2012

Adversarial Face Recognition and Phishing Detection Using Multi-Layer Data Fusion

Ramanathan, Venkatesh

http://hdl.handle.net/1920/8075

a service of Mason Publishing
ADVERSARIAL FACE RECOGNITION AND PHISHING DETECTION USING
MULTI-LAYER DATA FUSION

by

Venkatesh Ramanathan
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Computer Science

Committee:

Dr. Harry Wechsler, Dissertation Director
Dr. Jim X. Chen, Committee Member
Dr. Brent ByungHoon Kang, Committee Member
Dr. David A. Schum, Committee Member
Dr. Duminda Wijesekera, Committee Member
Dr. Sanjeev Setia, Department Chair
Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering

Date: Fall Semester 2012
George Mason University
Fairfax, VA
Adversarial Face Recognition and Phishing Detection Using Multi-Layer Data Fusion

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

by

Venkatesh Ramanathan
Master of Science
University of Maryland, 1994

Director: Harry Wechsler, Professor
Department of Computer Science

Fall Semester 2012
George Mason University
Fairfax
# TABLE OF CONTENTS

List of Tables ............................................................................................................. iv
List of Figures ............................................................................................................. v
List of Abbreviations ................................................................................................. vii
Abstract ..................................................................................................................... ix

1 Introduction ........................................................................................................... 1
  1.1 Motivation ......................................................................................................... 1
  1.2 Contributions ................................................................................................... 3
  1.3 Organization ..................................................................................................... 5

2 Identity Management ............................................................................................... 6
  2.1 Overview .......................................................................................................... 6
  2.2 Biometrics ......................................................................................................... 9
  2.3 Cybersecurity ................................................................................................... 11

3 Methods ................................................................................................................ 17
  3.1 Image Analysis ................................................................................................. 17
  3.2 Machine Learning ............................................................................................ 24
  3.3 Natural Language Processing .......................................................................... 31
  3.4 Performance Evaluation ................................................................................... 40

4 Face Recognition .................................................................................................... 44
  4.1 Motivation ......................................................................................................... 46
  4.2 Background ....................................................................................................... 48
  4.3 Adaptive and Robust Correlation Filters ......................................................... 61
    4.3.1 Methodology ............................................................................................... 61
    4.3.2 Architecture ............................................................................................... 73
    4.3.3 Experiments ............................................................................................... 77
  4.4 Anthropometric and Appearance-Based Recognition ....................................... 92
    4.4.1 Methodology ............................................................................................... 92
    4.4.2 Architecture ............................................................................................... 99
    4.4.3 Experiments ............................................................................................... 102
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.5 Summary</td>
<td>115</td>
</tr>
<tr>
<td>5 Phishing Detection</td>
<td>117</td>
</tr>
<tr>
<td>5.1 Motivation</td>
<td>117</td>
</tr>
<tr>
<td>5.2 Background</td>
<td>119</td>
</tr>
<tr>
<td>5.3 Phishing Email Detection</td>
<td>129</td>
</tr>
<tr>
<td>5.3.1 Methodology</td>
<td>130</td>
</tr>
<tr>
<td>5.3.2 Architecture</td>
<td>136</td>
</tr>
<tr>
<td>5.3.3 Experiments</td>
<td>142</td>
</tr>
<tr>
<td>5.4 Phishing Website Detection</td>
<td>158</td>
</tr>
<tr>
<td>5.4.1 Methodology</td>
<td>159</td>
</tr>
<tr>
<td>5.4.2 Architecture</td>
<td>162</td>
</tr>
<tr>
<td>5.4.3 Experiments</td>
<td>166</td>
</tr>
<tr>
<td>5.5 Impersonated Entity Discovery</td>
<td>172</td>
</tr>
<tr>
<td>5.5.1 Methodology</td>
<td>173</td>
</tr>
<tr>
<td>5.5.2 Architecture</td>
<td>185</td>
</tr>
<tr>
<td>5.5.3 Experiments</td>
<td>190</td>
</tr>
<tr>
<td>5.6 Summary</td>
<td>199</td>
</tr>
<tr>
<td>6 Conclusions</td>
<td>202</td>
</tr>
<tr>
<td>References</td>
<td>214</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1 ARCF Results (1) - Recognition (Hit) Rates for PCA, PCA+LDA and ARCF</td>
<td>89</td>
</tr>
<tr>
<td>Table 2 ARCF Results (2) - Recognition (Hit) Rates for PCA, PCA+LDA, and ARCF</td>
<td>90</td>
</tr>
<tr>
<td>Table 3 Ranking of Holistic Anthropometric Features</td>
<td>97</td>
</tr>
<tr>
<td>Table 4 Hybrid - Recognition (Hit) Rates for Disguised (Face with Beard) Images</td>
<td>107</td>
</tr>
<tr>
<td>Table 5 Hybrid - Recognition (Hit) Rates for Occluded (Half-Face) Images</td>
<td>107</td>
</tr>
<tr>
<td>Table 6 Hybrid - Improvements in Recognition Rates for Disguised and Occluded Images</td>
<td>108</td>
</tr>
<tr>
<td>Table 7 Datasets for Experiments Part-II</td>
<td>109</td>
</tr>
<tr>
<td>Table 8 Hybrid - Recognition (Hit) Rates: Train (Clean)/Test (Occluded-Sunglasses)</td>
<td>111</td>
</tr>
<tr>
<td>Table 9 Hybrid - Recognition (Hit) Rates: Train (Clean)/Test (Occluded-Scarf)</td>
<td>111</td>
</tr>
<tr>
<td>Table 10 Hybrid - Recognition (Hit) Rates: Train (Occluded-Sunglasses)/Test (Clean)</td>
<td>111</td>
</tr>
<tr>
<td>Table 11 Hybrid - Recognition (Hit) Rates: Train (Occluded-Scarf)/Test (Clean)</td>
<td>111</td>
</tr>
<tr>
<td>Table 12 phishGILLNET1 - PLSA Word/Topic Probabilities</td>
<td>149</td>
</tr>
<tr>
<td>Table 13 phishGILLNET1 - PLSA Model Performance</td>
<td>149</td>
</tr>
<tr>
<td>Table 14 phishGILLNET1 Classification Performance</td>
<td>150</td>
</tr>
<tr>
<td>Table 15 Classification Performance of SVM on Dataset Combination</td>
<td>150</td>
</tr>
<tr>
<td>Table 16 phishGILLNET2 3-Class (Phish/Spam/Good) Classification Performance</td>
<td>152</td>
</tr>
<tr>
<td>Table 17 phishGILLNET2 2-Class (Phish/Not-Phish) Classification Performance</td>
<td>152</td>
</tr>
<tr>
<td>Table 18 phishGILLNET3 Classification Performance</td>
<td>154</td>
</tr>
<tr>
<td>Table 19 phishGILLNET and Competing Methods Characteristic</td>
<td>156</td>
</tr>
<tr>
<td>Table 20 Performance Comparison – 3-Class Classification</td>
<td>157</td>
</tr>
<tr>
<td>Table 21 Performance Comparison - Binary Classification</td>
<td>157</td>
</tr>
<tr>
<td>Table 22 Phishing Website Detection - LDA Topics</td>
<td>169</td>
</tr>
<tr>
<td>Table 23 Phishing Website Detection - LDA Model Performance</td>
<td>169</td>
</tr>
<tr>
<td>Table 24 Phishing Website Detection - AdaBoost+LDA Classification Performance</td>
<td>170</td>
</tr>
<tr>
<td>Table 25 Phishing Website Detection - Classification Performance for Varying Phishing in Training</td>
<td>170</td>
</tr>
<tr>
<td>Table 26 Phishing Website Detection - State-of-the-art Comparison</td>
<td>172</td>
</tr>
<tr>
<td>Table 27 Impersonated Entity Detection Features</td>
<td>189</td>
</tr>
<tr>
<td>Table 28 Impersonated Entity Detection - LDA Performance</td>
<td>196</td>
</tr>
<tr>
<td>Table 29 Impersonated Entity Detection - Classification Performance</td>
<td>197</td>
</tr>
<tr>
<td>Table 30 Impersonated Entity Detection - CRF Performance</td>
<td>198</td>
</tr>
<tr>
<td>Figure</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>Figure 1 Face Authentication Stages</td>
<td>11</td>
</tr>
<tr>
<td>Figure 2 Phishing Attack Flow</td>
<td>16</td>
</tr>
<tr>
<td>Figure 3 Correlation Filters</td>
<td>21</td>
</tr>
<tr>
<td>Figure 4 Correlation Peaks and Peak Locations for the Same Subject Using Match Filters</td>
<td>63</td>
</tr>
<tr>
<td>Figure 5 Correlation Peaks and Peak Locations for the Same Subject with Distorted Parts</td>
<td>64</td>
</tr>
<tr>
<td>Figure 6 Correlation Peaks and Peak Locations for Different Subjects</td>
<td>65</td>
</tr>
<tr>
<td>Figure 7 Comparison of Correlation Filters</td>
<td>71</td>
</tr>
<tr>
<td>Figure 8 Effects of Both Temporal Change and Additive White Noise on MF, MACE, OTF, and ARCF</td>
<td>72</td>
</tr>
<tr>
<td>Figure 9 ARCF Architecture</td>
<td>74</td>
</tr>
<tr>
<td>Figure 10 ARCF Results for Disguise</td>
<td>81</td>
</tr>
<tr>
<td>Figure 11 ARCF Results for Scrambling of Face Parts</td>
<td>82</td>
</tr>
<tr>
<td>Figure 12 ARCF Results for Occlusion</td>
<td>83</td>
</tr>
<tr>
<td>Figure 13 ARCF Results for Varying Illumination</td>
<td>84</td>
</tr>
<tr>
<td>Figure 14 ARCF Results for Temporal Change and Varying Face Expression</td>
<td>85</td>
</tr>
<tr>
<td>Figure 15 ARCF Results - ROC Curves for PCA and PCA+LDA Disguised Faces</td>
<td>88</td>
</tr>
<tr>
<td>Figure 16 ARCF Results - ROC Curves (Basis-Clean, Training-Clean/Disguised, Testing-Clean/Disguised)</td>
<td>91</td>
</tr>
<tr>
<td>Figure 17 ARCF Results - ROC Curves (Basis-Clean+Occlusions, Training-Clean/Disguised, Testing-Clean/Disguised)</td>
<td>92</td>
</tr>
<tr>
<td>Figure 18 Holistic Anthropometric Features</td>
<td>96</td>
</tr>
<tr>
<td>Figure 19 Hybrid - ROC for Disguised Images</td>
<td>106</td>
</tr>
<tr>
<td>Figure 20 Hybrid - ROC for Occluded Images</td>
<td>107</td>
</tr>
<tr>
<td>Figure 21 Images (Clean and Occluded) from AR Face Database</td>
<td>109</td>
</tr>
<tr>
<td>Figure 22 Hybrid - ROC Curve for Occluded (Sunglasses) Images</td>
<td>112</td>
</tr>
<tr>
<td>Figure 23 Hybrid - ROC Curve for Occluded (Scarf) Images</td>
<td>113</td>
</tr>
<tr>
<td>Figure 24 Hybrid - ROC Curve for Clean Images (Train-Sunglasses)</td>
<td>114</td>
</tr>
<tr>
<td>Figure 25 Hybrid - ROC Curve for Clean Images (Train-Scarf)</td>
<td>115</td>
</tr>
<tr>
<td>Figure 26 Phishing Protection Techniques</td>
<td>120</td>
</tr>
<tr>
<td>Figure 27 phishGILLNET</td>
<td>131</td>
</tr>
<tr>
<td>Figure 28 Multi-Layered phishGILLNET</td>
<td>132</td>
</tr>
<tr>
<td>Figure 29 phishGILLNET - Parsed and TDF Matrix Builder</td>
<td>133</td>
</tr>
<tr>
<td>Figure 30 phishGILLNET1 Architecture</td>
<td>137</td>
</tr>
</tbody>
</table>
Figure 31 phishGILLNET1 - PLSA Model Trainer and Fold-In ........................................ 137
Figure 32 phishGILLNET2 Architecture ........................................................................ 140
Figure 33 phishGILLNET3 Architecture ........................................................................ 142
Figure 34 phishGILLNET1 Performance - Log Likelihood Versus Number of EM Steps .......................................................... 148
Figure 35 Phishing Website Detection Methodology ......................................................... 161
Figure 36 Phishing Website Detection Architecture .......................................................... 164
Figure 37 Impersonated Entity Detection Methodology .................................................... 174
Figure 38 Impersonated Entity Detection Architecture ..................................................... 186
Figure 39 Self-Managing Shield ...................................................................................... 209
# LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Artificial Intelligence</th>
<th>AI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Artificial Neural Network</td>
<td>ANN</td>
</tr>
<tr>
<td>Adaptive and Robust Correlation Filters</td>
<td>ARCF</td>
</tr>
<tr>
<td>Area Under Receiver Operating Characteristic</td>
<td>AUC</td>
</tr>
<tr>
<td>Correlation Filter</td>
<td>CF</td>
</tr>
<tr>
<td>Class Dependence Feature Analysis</td>
<td>CFA</td>
</tr>
<tr>
<td>Conditional Random Field</td>
<td>CMU</td>
</tr>
<tr>
<td>Carnegie Mellon University</td>
<td>CRF</td>
</tr>
<tr>
<td>Colorado State University</td>
<td>CSU</td>
</tr>
<tr>
<td>Distributed Associative Memories</td>
<td>DAM</td>
</tr>
<tr>
<td>Dynamic Programming</td>
<td>DLA</td>
</tr>
<tr>
<td>Dynamic Link Architecture</td>
<td>DP</td>
</tr>
<tr>
<td>Dynamic Space Warping</td>
<td>DSA</td>
</tr>
<tr>
<td>Elastic &lt;Bunch&gt; Graph Matching</td>
<td>E&lt;B&gt;GM</td>
</tr>
<tr>
<td>Elastic Graph Matching</td>
<td>EGM</td>
</tr>
<tr>
<td>Expectation Maximization</td>
<td>EM</td>
</tr>
<tr>
<td>Ensembles of Radial Basis Function</td>
<td>ERBF</td>
</tr>
<tr>
<td>Fusiform Face Area</td>
<td>FFA</td>
</tr>
<tr>
<td>Face Recognition Grand Challenge</td>
<td>FRGC</td>
</tr>
<tr>
<td>Fourier Transform</td>
<td>FT</td>
</tr>
<tr>
<td>Gram-Schmidt Linear Discriminant Analysis</td>
<td>GSLDA</td>
</tr>
<tr>
<td>Hyper Text Markup Language</td>
<td>HTML</td>
</tr>
<tr>
<td>Internet Protocol</td>
<td>IP</td>
</tr>
<tr>
<td>Kernel Class Dependence Feature Analysis</td>
<td>KCFA</td>
</tr>
<tr>
<td>Kernel Discrete Cosine Transform</td>
<td>KDCT</td>
</tr>
<tr>
<td>Linear Discriminant Analysis</td>
<td>LDA</td>
</tr>
<tr>
<td>Latent Dirichlet Allocation</td>
<td>LDA</td>
</tr>
<tr>
<td>Line Edge Map</td>
<td>LEM</td>
</tr>
<tr>
<td>Latent Semantic Analysis</td>
<td>LSA</td>
</tr>
<tr>
<td>Minimum Average Correlation Energy Filter</td>
<td>MACE</td>
</tr>
<tr>
<td>Match Filter</td>
<td>MF</td>
</tr>
<tr>
<td>Multipart Internet Mail Extension</td>
<td>MIME</td>
</tr>
<tr>
<td>Machine Learning</td>
<td>ML</td>
</tr>
<tr>
<td>Minimum Variance Synthetic Discriminant Function Filter</td>
<td>MVSDSF</td>
</tr>
<tr>
<td>Named Entity Recognition</td>
<td>NER</td>
</tr>
<tr>
<td>National Institute of Standards and Technology</td>
<td>NIST</td>
</tr>
<tr>
<td>Term</td>
<td>Abbreviation</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Natural Language Processing</td>
<td>NLP</td>
</tr>
<tr>
<td>Optimal Trade-off Filter</td>
<td>OTF</td>
</tr>
<tr>
<td>Principal Component Analysis</td>
<td>PCA</td>
</tr>
<tr>
<td>Phishing Identification by Learning on Features of Email Received</td>
<td>PILFER</td>
</tr>
<tr>
<td>Probabilistic Latent Semantic Analysis</td>
<td>PLSA</td>
</tr>
<tr>
<td>Part of Speech</td>
<td>POS</td>
</tr>
<tr>
<td>Phase Only Unconstrained Minimum Average Correlation Energy Filter</td>
<td>POUMACE</td>
</tr>
<tr>
<td>Peak to Side Lobe Ratio</td>
<td>PSR</td>
</tr>
<tr>
<td>Synthetic Discriminant Function Filter</td>
<td>SDF</td>
</tr>
<tr>
<td>Short Message Service</td>
<td>SMS</td>
</tr>
<tr>
<td>Singular Value Decomposition</td>
<td>SVD</td>
</tr>
<tr>
<td>Term Document Frequency</td>
<td>TDF</td>
</tr>
<tr>
<td>Tempered Expectation Maximization</td>
<td>TEM</td>
</tr>
<tr>
<td>Uniform Resource Locator</td>
<td>URL</td>
</tr>
<tr>
<td>Voice Over Internet Protocol</td>
<td>VOIP</td>
</tr>
</tbody>
</table>
This thesis addresses digital identity for biometric / face recognition screening and cyberspace security subject to denial and deception characteristic of adversarial behavior. The adversarial aspect concerns defense and offense operations that involve impostors and identity theft. Denial and deception correspond to occlusion and disguise for biometrics, while for cyberspace security they correspond to spoofing and obfuscation. To prevent or mitigate the impacts of adversarial behavior from offensive attacks this thesis proposes the use of multi-layer data fusion. Multi-layer aspect of fusion refers to features, representations, algorithms, decision-making, adversarial aspects and their purposeful combinations. This novelty, feasibility, and utility of our research is illustrated in the physical and cyber worlds: (i) robust face recognition in the presence of occlusion and disguise, and (ii) phishing detection to prevent identity theft through spoofing and obfuscation.
The novel face recognition methodologies include: (i) Adaptive and Robust Correlation Filters (ARCF) built around match filters and recognition-by-parts, and (ii) hybrid anthropometric and appearance based biometric authentication using boosting for feature level fusion and backpropagation learning for decision level fusion. The cluster and strength of the ARCF correlation peaks indicate the confidence in the face authentications. Experimental evidence using the AR benchmark database shows that our methods are highly reliable in the presence of occlusion, disguise, and illumination, expression and temporal variability.

The novel phishing detection methodologies address: (i) phishing email detection using semantic topics and Probabilistic Latent Semantic Analysis (PLSA), boosting, and Co-Training for both labeled and unlabeled examples, (ii) phishing website detection using Latent Dirichlet Allocation (LDA) and boosting, and (iii) impersonated entity discovery using LDA, boosting, and Condition Random Field (CRF). The phishing detection methodology handles the adversarial use of synonyms, polysemy (words with multiple meanings) and other linguistic variations. In addition, the same methodology requires only a small percentage of data to be annotated thus saving time, labor, and avoiding errors incurred during human annotation. The phishing website detection methodology is device and language neutral. The impersonated entity discovery methodology automatically extracts the entity the attacker is trying to spoof. This helps service providers to collaborate with each other to exchange attack information and protect their customers. Experimental results on SPAM Archive, which is one of the largest public
corpus, show that our phishing detection methodology outperforms state of the art phishing detection methods.
1 INTRODUCTION

This chapter presents the motivation for this research work, research contributions, and the organization of this dissertation. Background information on the growing crime in the cyberspace, namely, identity theft, the attackers motivation, and the technique employed by attackers are presented in 1.1. The methodology and contributions of this research are presented in 1.2, followed by the organization of this dissertation in section 1.3.

1.1 Motivation

Stealing a person’s identity is one of the most profitable crimes committed by criminals. In the United States, more than 9 million cases of identity theft have been reported in 2011 costing billion of dollars [1]. Identity theft has been around for many years while the means of committing that crime has changed with technology. The traditional way criminals steal a person’s identity is by killing the individual. Another way to steal identity is using phone scams where criminals inform some person that they have won a sweepstake, and convince him/her to reveal some personal information in order to claim the money. Dumpster diving is another way to steal identity. When people discard letters, financial records, and other personal information in the garbage dump without shredding, criminals scavenge those dumps looking for sensitive information.
such as credit card, bank account and social security numbers, and, use that information to commit crimes.

In the cyberspace, criminals steal cyber identity using (i) social engineering attacks such as phishing, vishing and SMiShing, (ii) installing spyware, key logger and malware in user’s machines, and, (iii) by stealing sensitive data by exploiting software vulnerabilities. The most common social engineering attack is phishing. Like in traditional fishing where fishermen troll the river in a boat to catch fish, in phishing, attackers troll the internet using email message with convincing content as baits to steal users personal information. The email directs the user via a hyperlink to a website owned by criminals that looks very similar to a legitimate website. The user will then be asked to enter personal and financial information either to update existing information or to purchase a product. In reality this lets criminal to have access to that valuable information which they then use to commit other fraud or sell to a bidder. This technique has been around since 1996 but is becoming more common and more and more sophisticated. Vishing (voice + phishing) is the criminal practice of using social engineering tactics over the voice over internet protocol (VOIP) telephone system. Attackers spoof the caller identification of financial institutions and collect credit card and other personal information using automated voice messaging. SMiShing (SMS + phishing) uses cell phone text messaging to deliver the bait to users. The bait is usually a URL to attacker’s web site.

One way to counteract cyber identity theft is to enforce better authentication mechanisms than password. Even when credentials are stolen, use of the stolen
credentials should not be effective in the hands of criminals. Biometric technology helps
with recognition of people using physiological characteristics (face, fingerprint) and/or
behavioral characteristics (signature, voice). Biometrics provides greater security and
convenience than traditional authentication schemes. While passwords, tokens, and PINs
can be lost, stolen, or forgotten, the same is not the case using biometrics, as the
authentication system takes biological input directly from the user. Towards the goal of
providing a better and secure authentication mechanism, thereby making the existing
attacks that steal cyber identities and passwords ineffective, this research develops robust
face recognition methodologies for human authentication.

To protect users from falling for identity theft attacks also requires a secure
cyberspace so that attacks are stopped before it gets to the end user. As such attacks are
sophisticated and infeasible for humans to recognize and not fall for them, there is a need
for robust detection methodologies that can be employed to automatically detect and stop
such attacks. Towards the goal of preventing one such attack, phishing, from reaching the
end user, this research develops robust content driven phishing detection methodologies
which combines the power of natural language processing and machine learning.

1.2 Contributions

The main contributions of this research are as follows:

(i) Adaptive and Robust Correlation Filters (ARCF) face recognition
methodology, which employs recognition-by-parts strategy. ARCF provide information
that involves both appearance and location. The cluster and strength of the ARCF
correlation peaks indicate the confidence of the face authentication made, if any. The
development of ARCF, motivated by MACE filters (see Chapter 3) and adaptive beam-forming from radar/sonar, is driven by Tikhonov regularization (see Chapter 4). The adaptive aspect of ARCF comes from their derivation using both training and test data, similar to transduction. ARCF is suitable for face recognition even in the presence of occlusion, disguise, temporal and expression variations.

(ii) Holistic anthropometric and appearance based face recognition methodology, which employs feature level fusion using boosting and decision level fusion using backpropagation. In addition to standard head and face geometric measurements, the proposed anthropometric features include additional measurements below the face to describe the neck and shoulder and their contextual relations to head and face. The appearance-based features include standard Eigenfaces or Fisherfaces. The proposed methodology can train on clean data and authenticate on occluded/disguised data, or train on occluded/disguised data and authenticate on clean data.

(iii) Phishing email detection methodology, which employs Probabilistic Latent Semantic Analysis (PLSA), AdaBoost and Co-Training. The methodology handles synonyms (multiple words with similar meanings), polysemy (words with multiple meanings), intentionally misspelled words and other linguistic variations found in phishing. In addition, the methodology requires only a small percentage of data be annotated thus saving time, labor, and avoiding errors incurred in human annotation.

(iv) Phishing website detection methodology, which employs Latent Dirichlet Allocation (LDA) and AdaBoost. The content driven methodology is robust to changes in
word usage. It is device neutral (can be applied to desktop and mobile devices) and language neutral (can be applied to content in different languages).

(v) Impersonated entity discovery methodology, which employs LDA, AdaBoost and Condition Random Field (CRF). The methodology automatically extracts the entity the attacker is trying to portray. This helps service providers to collaborate with each other to exchange attack information and protect their customers. The legitimate organization can take down the offending site thus preventing its customers from falling for phishing, which in turn leads to satisfied customers.

1.3 Organization

The rest of the dissertation is organized as follows:

Chapter 2 – Discusses identity management systems with specific emphasis on two major aspects, biometrics and cyber security.

Chapter 3 – Presents the overview of methods employed here, namely, image analysis, machine learning, natural language processing and performance evaluation.

Chapter 4 – Presents robust face recognition methodologies which employs image analysis and machine learning, to build authentication systems that account for occlusion and disguise.

Chapter 5 – Presents multi-layered content driven phishing detection methodologies that employs natural language processing and machine learning.

Chapter 6 – Concludes summarizing major novelties of this research and suggestions for future research.
As the goal of this research is to develop methodologies to protect users from identity theft, this chapter discusses identity management and key aspects such as biometrics and cybersecurity. An overview of identity management is presented in section 2.1, followed by biometrics in section 2.2, and cybersecurity aspects in section 2.3.

2.1 Overview

Identity is defined as information that can be used to uniquely identify, contact, or locate a single person or can be used with other sources to uniquely identify a single individual [2]. It is also commonly referred in information security as Personally Identifiable Information (PII). Verifying the identity of a person or entity that seeks access to resources in the cyberspace (or) that authors an electronic document (or) that signs an electronic document, is the domain of what has come to be called identity management (IdM). In the physical word, the information to identify an individual includes name, physical appearance, government issued documents such as birth certificate, drivers license, passport and social security number, documents issued by financial institutions such as credit card, bank card, etc. In the cyberspace, this information includes email address, login name, tokens, cookies, secure sockets layer (SSL) certificates, radio frequency identification tag identifiers, IP addresses, etc. The
fundamental difference between the physical and the cyber space is that cyber identity is not required to belong to the same physical identity. For example, an individual can use the login name of his or her child or his spouse as long as the individual has the secure credentials. This opens the possibility of abuse by hackers to use stolen cyber identity of other individuals. Also, an individual can have many cyber-identities such as one for an email account, one for online banking, one for a e-commerce website, and so on, whereas the same individual has only one social security number. Hence, in order to apply the physical world security to the cyberspace to prevent abuse by attackers, there is a need to enhance the cyber identity with some physically identifiable information, e.g., biometrics.

The identity management system has 8 core functions namely, authorization, authentication, user registration and enrollment, password management, auditing, user self-service, central administration, and delegated administration [2]. When a user initiates a request for access to a resource, the identity management first authenticates the user by asking for credentials, which may be in the form of a username and password, digital certificate, smart card, or biometric data. After the user successfully authenticates, the identity management system authorizes the appropriate amount of access based on the user's identity and attributes. The access control component will manage subsequent authentication and authorization requests for the user, which will reduce the number of passwords the user will have to remember and reduce the number of times a user will have to perform a login function. The identity management system allows users to register accounts with the identity management system and also to enroll for access privileges to a particular resource. If the user cannot authenticate with the identity
management system the user will be provided the opportunity to register an account. Once an account is created and the user successfully authenticates, the user must enroll for access privileges to requested resources. The enrollment process may be automated based on set policies or the owner of the resource may manually approve the enrollment. Only after the user has successfully registered with the identity management system and enrolled for access will access to that resource be granted. The identity management system allows for password management. Users are able to reset their own passwords and synchronize passwords across multiple systems. The identity management system facilitates auditing of user and privilege information. The identity management system can be queried to verify the level of user privilege. The identity management system provides data from authoritative sources, providing auditors with accurate information about users and their privileges. The identity management system allows users to maintain their own personnel information and perform certain routine account tasks. For example, users can update their personal contact information, change their passwords, or synchronize passwords across all systems. If necessary, the changes can be validated before the appropriate authoritative sources are updated. The identity management system allows administrators to centrally manage multiple identities. Administrators can centrally manage both the content within the identity management system and the structural architecture of the identity management system. The identity management system allows delegated administration, so that administrators can manage identities for which they are responsible. Delegated administrators are not able to make any structural changes to the identity management system. Delegated administrators are only able to
manage the information stored in the identity management system. One of the major focus of this research is on the use of biometrics for the authentication function of the identity management system. An overview of biometrics will be presented next.

2.2 Biometrics

Authentication is essentially performed by one of three means, (i) something the person knows (e.g., a secret such as a PIN, password or other secret code), (ii) something the person possesses (e.g., a cryptographic key, an ATM card, a smart card, drivers license, or other physical token), (iii) something the person is (e.g., biometric characteristic). Biometrics is the technology to recognize human being using their physiological and behavioral characteristics. Physiological characteristics are related to part of the human body while the behavioral characteristics are related to mannerisms. Examples of physiological characteristics include face, hand, eye, ear, fingerprints and DNA while those of behavioral characteristics include signature, voice and movement. The major focus of this research is face authentication, which will be discussed next.

2.2.1 Face Authentication

In face authentication, an individual’s face is scanned to extract data that includes coordinates of characteristic points that outline the face. Distance between points and angles, skin color, curvature etc., are used for authentication. It consists of three main stages (Figure 1). During the enrollment phase, an individual’s face is first scanned by a biometric reader, which produces a digital representation. The data capture during the enrollment process may or may not be supervised by a human depending on the application. A quality check is generally performed to ensure that the acquired sample
can be reliably processed by successive stages. In order to facilitate matching, the input
digital representation is further processed by a feature extractor to generate a compact but
expressive representation called a template. Depending on the application, the template
may be stored in the central database of the biometric system or be recorded on a smart
card issued to the individual. Usually, multiple templates of an individual are stored to
account for variations observed in the biometric trait and the templates in the database
may be updated over time. In the verification phase, the system validates a person’s
identity by comparing the captured biometric data with his or her own biometric
template(s) stored in the system database. In such a system, an individual who desires to
be recognized claims an identity, usually via a personal identification number (PIN), a
user name, or a smart card, and the system conducts a one-to-one comparison to
determine whether the claim is true or not. Identity verification is typically used for
positive recognition, where the aim is to prevent multiple people from using the same
identity. In the identification mode, the system recognizes an individual by searching the
templates of all the users in the database for a match. Therefore, the system conducts a
one-to-many comparison to establish an individual’s identity (or fails if the subject is not
enrolled in the system database) without the subject having to claim an identity. An
overview of cybersecurity, the other major focus of this research, will be discussed next.
2.3 Cybersecurity

Cybersecurity encompass technologies, processes and practices designed to protect networks, computers, programs and data from attack, damage or unauthorized access. The Homeland Security Administration predicts a growing number cyber attacks is the biggest threat to nations security and will cripple economy. In order to protect our nation from cyber attacks, according to a December 2010 analysis of U.S. spending plans, the federal government has allotted over $13 billion annually over the next five years [3]. Cybersecurity consists of following four essential components, namely, Application Security, Information Security, Network Security, and Data Security.

2.3.1 Application Security
The use of software, hardware and procedural methods to secure applications from security threat is defined as Application Security. In earlier days security was not built into application instead added as an afterthought after attackers exposed security flaws costing corporations billions of dollars. Nowadays, security is becoming increasingly important to be considered in the development of an application from start. Application should be built in such a way that the chance of attackers to access, steal and modify resources that application serves must be minimal to nonexistent. The set of actions that are built into the application to protect resources are termed counter measures. Examples of counter measures include encryption and decryption, authentication systems, anti-virus, anti-spyware and anti-malware programs.

2.3.2 Information Security

Information Security is about protecting information from unauthorized access, use, disclosure, modification, inspection, recording or destruction. In other words, Information Security means adhering to the Confidentiality, Integrity, and Availability (CIA) principle of protecting and securing information. Confidentiality prevents disclosure of information to unauthorized individuals. Integrity implies the data cannot be modified without detection. Availability means the system must be available for access when in need. The CIA principle has later been expanded to include Authenticity and Non-repudiation. Authenticity ensures data, communications, and transactions are genuine. Non-repudiation means an individual or an entity cannot deny having sent or received information. These five guiding principles ensure information is protected and
secured from unauthorized access. To protect information from security breach requires continuous training, monitoring, detection, protection, incident response and repair.

2.3.3 Network Security

Network Security involves protecting networks and network attached resources from unauthorized access. Networks include both public networks (internet) and private networks (company network). The job of securing the network lies with network administrators. Access to a network is typically controlled by authenticating the user using credentials such as passwords or tokens. Once the user is authenticated, firewall enforces access policies such as the resources (servers, documents, databases) the authenticated user can view/modify. An anomaly detection system typically monitors network for intrusions or attacks using data collected at the network level. A decoy such as honeypot may also be deployed in the network as a surveillance tool to detect impending attacks.

2.3.4 Data Security

Data Security means protecting a database from destructive forces and the unwanted actions of unauthorized users. Software based security solutions encrypt the data to prevent data from being stolen. However, a malicious program or a hacker may corrupt the data in order to make it unrecoverable or unusable. Similarly, encrypted operating systems can be corrupted by a malicious program or a hacker, making the system unusable. Hardware-based security solutions can prevent read and write access to data and hence offers very strong protection against tampering and unauthorized access. Backups are used to ensure data, which is lost, can be recovered. Data masking of
structured data is the process of obscuring (masking) specific data within a database table or cell to ensure that data security is maintained and sensitive information is not exposed to unauthorized personnel. Data erasure is a method of software-based overwriting that completely destroys all electronic data residing on a hard drive or other digital media to ensure that no sensitive data is leaked when an asset is retired or reused.

The constant evolving nature of security risks requires incorporating the above security components in any organization. The traditional approach has been to focus most resources on the most crucial system components and protect against the biggest known threats. This resulted in some less important system components unprotected and some less dangerous risks not protected against. This approach is not sufficient in the current environment. To deal with the current environment, a more proactive and adaptive approach is required. The National Institute of Standards and Technology (NIST) guideline recommends a shift towards continuous monitoring and real-time assessments. The focus of this research is to develop such proactive solution for one such cyber attack, phishing, which will be presented next.

2.3.5 Phishing

Phishing is a criminal activity where attackers aim to steal personal and financial information from innocent victims. Like in traditional fishing where fishermen troll the river in a boat to catch fish, in phishing, attackers troll the internet using email messages with convincing content as bait to steal users personal information. It is infeasible for humans to recognize the bait, as they look authentic. Spear phishing is a dangerous variation of phishing, which is much more personalized. It is more dangerous than
regular phishing because the sender knows exactly whom users do business with, who user’s friends are, and what kind of accounts users have. Instead of sending out hundreds of emails randomly hoping a few victims will fall for the bait, spear phishers target select groups of people with something in common e.g., they work at the same company, bank at the same financial institution, attend the same college, order merchandise from the same website, etc. The emails are usually sent from organizations or individuals the potential victims would normally get emails from, making them even more deceptive. Phishing has been around since 1996 but is becoming more common and more and more sophisticated.

A phishing attack generally works as follows (Figure 2). First, criminals need some inside information on their targets to convince them that the emails are legitimate. They often obtain it by hacking into an organization’s computer network or sometimes by combing through other websites, blogs, and social networking sites such as Facebook, Twitter etc. Then, they send emails that look like the real thing to targeted victims, offering all sorts of urgent and legitimate sounding explanations as to why they need your personal data. Finally, the victims are asked to click on a link inside the email that takes them to a fake but realistic looking website, where they are asked to provide passwords, account numbers, user IDs, access codes, etc. Once criminals have your personal data, they can access your bank account, use your credit cards and create a whole new identity using your information.
Phishers can also trick users into downloading malicious software after they click on a link embedded in the email. This is a useful tool in crimes like economic espionage where sensitive internal communications can be accessed and trade secrets stolen.

Malware can also hijack user’s computer, and hijacked computers can be organized into enormous networks called botnets that can be used for denial of service attacks. The anti-phishing working group reports that the phishing problem has grown significantly over the years. Existing protection methods fail to completely stop the attack. The goal of this research is to develop robust methodologies to stop phishing before it gets to the user.

Figure 2 Phishing Attack Flow
3 METHODS

The goal of this research is to develop two functional components that form the building blocks of an identity management system, face recognition and phishing detection. Face recognition employs image analysis and machine learning. Phishing detection employs machine learning and natural language processing. A review of these methods is presented in this chapter.

3.1 Image Analysis

Image analysis is the process of extracting meaningful information from images using various image processing techniques for the purpose of identification and recognition. In this research, four types of methods are employed, namely, Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Fisherfaces (PCA+LDA) and Correlation Filters. These methods are presented next.

3.1.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is one of the most successful methods that have been used in image recognition and compression. PCA is a statistical method under the broad title of factor analysis. The purpose of PCA is to reduce the large dimensionality of the data space (observed variables) to the smaller intrinsic dimensionality of feature space (uncorrelated variables), which is needed to describe the
data economically. PCA can be employed for prediction, redundancy removal, feature extraction, data compression, etc.

PCA works as follows [4]. A two dimensional facial image can be represented as a 1-dimensional vector by concatenating each row/column into a vector. If we have $M$ vectors of size $N$ representing a set of sampled images and $p_j$ are the pixel values, the 1-dimensional vector is represented as:

**Equation 1**

$$x_i = [p_1...p_N]^T, i = 1,...,M$$

The images are mean centered by subtracting the mean image from each image vector. Let $w_i$ be defined as the mean centered image:

**Equation 2**

$$w_i = x_i - m$$

The goal is to find a set of $M$ orthonormal vectors $e_i$ for which the following quantity $\lambda_i$ is maximized:

**Equation 3**

$$\lambda_i = \frac{1}{M} \sum_{n=1}^{M} (e_i^T w_n)^2$$

e_i’s and $\lambda_i$ are the eigenvectors and eigenvalues respectively of the covariance matrix:

**Equation 4**

$$C = WW^T$$

where, $W$ is a matrix composed of the column vectors $w_i$ placed side by side. The eigenvectors (also called as eigenfaces) corresponding to nonzero eigenvalues of the covariance matrix produce an orthonormal basis for the subspace within which most
image data can be represented with a small amount of error. The eigenvector associated with the largest eigenvalue is the one that reflects the greatest variance in the image. Once eigenfaces have been computed, face identification and verification tasks can be performed. To identify an unknown image, that image is projected onto the face space to obtain its set of weights. By comparing set of weights for the unknown face to sets of weights of known faces, the face can be identified. We employ PCA to compare results of our Adaptive and Robust Correlation Filters (ARCF) (see section 4.3) and to develop the hybrid anthropometric and appearance based face recognition (see section 4.4).

3.1.2 Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA) finds a coordinate face space that best discriminates among classes [5]. For all samples of all classes, the between-class scatter matrix $S_B$ and the within-class scatter matrix $S_W$ are defined as:

**Equation 5**

$$S_B = \sum_{i=1}^{c} M_i (x_i - \mu)(x_i - \mu)^T$$

**Equation 6**

$$S_W = \sum_{i=1}^{c} \sum_{x_k \in X_i} (x_k - \mu_i)(x_k - \mu_i)^T$$

where, $M_i$ is the number of training samples in class $i$, $c$ is the number of distinct classes, $\mu_i$ is the mean vector of samples belonging to class $i$ and $X_i$ represents the set of samples belonging to class $i$ with $x_k$ being the $k$-th image of that class. $S_W$ represents the scatter of features around the mean of each face class and $S_B$ represents the scatter of features around the overall mean for all face classes. The goal is to maximize $S_B$ while minimizing
\( S_W \), in other words, maximize the ratio \( \det[S_B] / \det[S_W] \). This ratio is maximized when the column vectors of the projection matrix \( W_{LDA} \) are the eigenvectors of \( S_W^{-1} S_B \).

### 3.1.3 Fisherfaces (PCA+LDA)

Fisherfaces is a technique that was developed to overcome the problem of \( S_W \) from becoming singular. Fisherfaces avoids this problem by projecting the image set to a lower dimensional space so that the resulting within-class scatter matrix \( S_W \) is nonsingular. This is achieved by using PCA to reduce the dimension of the feature space and, then, applying the standard LDA to reduce the dimension to \( c-1 \). Fisherfaces is employed for comparing the performance of ARCF (see section 4.3) and for developing the hybrid face recognition using anthropometric features and appearance-based features (see section 4.4).

### 3.1.4 Correlation Filters

In this section a brief overview of various correlation filters (CF) (Figure 3). Face is authenticated by cross- correlating the input face with a synthesized template or filter and processing the resulting correlation output. Cross-correlation is implemented efficiently using Fast Fourier Transforms (FFTs). The correlation output is searched for peaks. The relative heights of these peaks are used to determine whether the face of interest is present or not. Correlation filters form the basis for ARCF (see section 4.3).

We start with the simple match filter (MF) and end with the Optimal Trade-off Filter (OTF). The following convention on notation is followed: vector \( b \) will be in lower case and bold, matrix \( B \) will be upper case bold, and scalar \( b \) will be in lower case. \( s \) refers to a training vector, \( n \) to additive noise, \( x \) to the test vector, \( d \) to the desired
response vector, and \( h \) to the filter weight. \( D_x \) refers to a matrix containing the power spectrum of \( x \) on the diagonal and zeros elsewhere, and \( H \) refers to the conjugate transpose.

![Flowchart of Correlation Filters](image)

**Figure 3 Correlation Filters**

**Match Filter (MF)**

The Match Filter (MF) shown below is optimal against white noise but allows training with only one exemplar \( s \):

**Equation 7**

Minimize: \( h^H h \)

Subject to \( s^H h = d \)

Solution: \( h = s(s^H s)^{-1} d \)
Synthetic Discriminant Function Filter (SDF)

To train with multiple exemplars, one has to use the Synthetic Discriminant Function Filter (SDF), which is robust to white noise.

**Equation 8**
Minimize \( h^H h \)

Subject to \( S^H h = d \)

where \( S = [s_1 \ldots s_M] \) and \( d = 1_M \) (vector of \( M \) ones)

**Solution:** \( h = S(S^H S)^{-1}d \)

Minimum Variance Synthetic Discriminant Function Filter (MVSDF)

Robustness to general non-white noise leads to the Minimum Variance Synthetic Discriminant Filter (MVSDF) [6] described below:

**Equation 9**
Minimize \( h^H Q_n h \)

Subject to \( S^H h = d \)

where \( S = [s_1 \ldots s_M] \) and \( d = 1_M \)

**Solution:** \( h = Q_n^{-1} S(S^H Q_n^{-1} S)^{-1} d \)

with \( Q_n \) defined as a diagonal matrix containing the average training noise power spectrum. When the noise power spectrum is not known, it is typically assumed to be spectrally white, \( Q_n = \sigma_n^2 I \), and the MVSDF filter reduces to the SDF filter. Like SDF and MF, MVSDF suffers from the presence of side-lobes, which are secondary peaks away from the true correlation peak on the correlation surface, even when training and testing with the same image.
Minimum Average Correlation Energy Filter (MACE)

This problem due to secondary peaks is addressed using the Minimum Average Correlation Energy Filter (MACE) [7] described below:

**Equation 10**
Minimize $h^H D_s h$

Subject to $S^H h = d$

where $S = [s_1 \ldots s_M]$ and $d = 1_M$

Solution: $h = D_s^{-1} S (S^H D_s^{-1} S)^{-1} d$

with $D_s$ defined as a diagonal matrix containing the average training exemplars’ power spectrum. The MACE filter minimizes the correlation side-lobes for the training images. It is, however, extremely sensitive to the noise that is typically present in the test image.

Optimal Trade-off Filter (OTF)

To improve the MACE filter robustness to noise and distortion, the Optimal Trade-off filter (OTF) filter [8] was developed and is described below:

**Equation 11**
Minimize $h^H D_s h, h^H Q_n h$

Subject to $S^H h = d$

where $S = [s_1 \ldots s_M]$ and $d = 1_M$

Solution: $h = T^{-1} S (S^H T^{-1} S)^{-1} d$

with $T = (\alpha D_s + \sqrt{1 - \alpha^2} Q_n)$. OTF was designed to trade-off, using $\alpha$, between correlation side-lobe suppression and noise robustness. $\alpha = 1$ leads to the MACE filter (maximum side-lobe suppression), while $\alpha = 0$ yields the MVSD filter (maximum noise
robustness). = 0 with white the noise assumption yields the SDF filter. OTF has been
used on face verification [9].

3.2 Machine Learning

Machine learning (ML) is a discipline that involves the design and development
of algorithms that enable machines to change their behavior based on observed data.
Machine learning belongs to the field of artificial intelligence (AI). A machine or a
system is considered to learn from experience if the performance of tasks improves with
experience. The goal of the learner is to have generalization capability. Given training
data containing inputs and outputs (labels), ML seeks model that can predict the output of
new unseen data.

Inputs, also known as the feature vector or attributes, to machine learning
algorithms can belong to three main categories. They can be real values or discrete values
or categorical values. Categorical values can in turn be ordered or unordered values. A
special case has boolean values, which has values of 1 or 0 in case of discrete, and true or
false incase of categorical variable. Outputs can be real numbers or categorical values. If
the output is categorical, then the algorithm is termed a classifier. If it is a real value, it is
termed a function estimator. If the output is a boolean value, the learning methodology is
called as concept learning and the function that is being learned is called a concept. There
are several ways to perform training. In a batch mode, the entire training data is used at
once to derive the function. In incremental training, one example is selected at a time and
this is used to alter the current hypothesis. This process is repeated iteratively. Another
method of training is random method, with and without replacement. In this procedure,
an example or a set of examples is selected at random from the training set and the process is continued iteratively. In online method, the training examples are used when they become available. The major types of machine learning algorithms and their applications in this research are presented next.

3.2.1 Supervised Learning

In supervised learning the training examples consists of input vector and a desired output value (also called the label). The algorithm analyzes the training data and produces a function. In the case of discrete values for output, the function is called a classifier while in case of real values it is called regression. The function should be able to predict the output of any unseen data. We employ two main supervised learning methods, Backpropagation – for face recognition (see Chapter 4), and, AdaBoost – for face recognition (see Chapter 4) and phishing detection (see Chapter 5). These methods are described next.

**Backpropagation**

Backpropagation is a supervised learning method of training an artificial neural network [10]. The network is defined by, an input layer, an output layer and a variable number of hidden layers. Each layer is composed of one or more neurons. Each neuron in a layer is connected to each neuron of the previous and the next layers. The connection between two neurons is parameterized by a weight. A value for the weight is estimated during the training process so as to minimize the difference between the computed output of the network and the targeted output. Each individual neuron is connected to an
additional weight independent of any inter-layer connection. This additional weight plays the role of an offset or bias.

Training the backpropagation network works as follows. The weights are initialized to random numbers ranging between -1 and 1. The neural network is trained by iteratively adjusting the weights of the connections in such a way that the error function is minimized. The training algorithm, called the delta rule, is a steepest descent algorithm, which, at each iteration, updates the weights by propagating the error into the network’s weights. Algorithm requires several iterations before reaching an acceptable solution. When the network is trained, the network is read to generalize with new data sets for prediction or classification. The backpropagation method is employed to perform the decision level fusion of anthropometric and appearance-based face recognition (see Chapter 4).

**AdaBoost**

To cope with data variability, one employs classifier ensemble. The idea behind classifier ensemble is to combine predictions of multiple classifiers and produce a single classifier. The prediction result from the combined classifier is generally better than that of individual classifiers. Results from an ensemble are less dependent on strangeness of employing a single training set and thus it reduces bias and variance. There are several ways of forming an ensemble or a collection. The two most popular ones are bagging and boosting. Both these methods rely on re-sampling of the data to obtain different training sets for each of the classifiers. Here, we employ the boosting technique, specifically, AdaBoost [11].
The idea behind AdaBoost, developed by Freund and Schapire [11], is to produce a series of classifiers. The training data used for each member of the series is chosen based on the performance of earlier classifiers in the series. Incorrectly predicted examples are selected more frequently than correctly predicted examples. Thus, boosting produces classifiers that are better in prediction than the current ensemble. Unlike bagging, AdaBoost considers performance of the earlier classifiers. The algorithm is detailed as follows:

Given input training data \((x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\), where \(x_i\) belongs to feature space \(X\) and \(y_i\) belongs to label set \(Y = \{-1, +1\}\),

Step 0: Initialize weights for the first iteration, \(D_1(i) = 1/m\)

For iteration index \(t = 1, \ldots, T\), where \(T\) is the number of iterations,

Step 1: Train a weak learner using distribution \(D_t\),

Step 2: Obtain weak hypothesis:

\(h_t : X \rightarrow \{-1, +1\}\)

with error:

\[\varepsilon_t = \sum_{i: h_t(x_i) \neq y_i} D_t(i).\]

Step 3: Compute:

\[\alpha_t = \frac{1}{2} \ln \left( \frac{1 - \varepsilon_t}{\varepsilon_t} \right).\]

Step 4: Updates weights for this step:

\[D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}\]

\(Z_t\)
where $Z_t$ is the normalization factor. The final strong classifier, which is a weighted majority of $T$ weak hypothesis, is given as:

$$H(x) = \text{sign} \left( \sum_{t \in T} \alpha_t h_t(x) \right).$$

Both face recognition (see Chapter 4) and phishing detection (see Chapter 5) employs AdaBoost using several learning algorithms such as C4.5 decision trees [12], rule based classifier RIPPER [13], random forest [14], support vector machines [15], logistic regression [16] and artificial neural networks [17] to build robust classifiers.

### 3.2.2 Unsupervised Learning

In unsupervised learning, training examples do not have a label (output value). The algorithm tries to extract the hidden structure from the training data. This learning method is used for summarization and partitioning of the data that share common features. The most common unsupervised learning technique is clustering. Some examples of clustering algorithm include k-means, mixture models and hierarchical clustering. Self-organizing map and adaptive resonance theory are approaches that employ unsupervised learning. The topic models discussed later on (see Section 3.3) for topic discovery belong to this type of learning.

### 3.2.3 Semi-supervised Learning

The training examples in semi-supervised learning include both labeled and unlabeled data. The proportion of unlabeled data is typically much higher than that of labeled data. By combining both labeled and unlabeled data, this technique is known to result in better performance than just using supervised or unsupervised learning. In real world, it is much easier and cheaper to obtain unlabeled data. Labeling requires humans
who are experts in the field to manually annotate. This process is very expensive and time consuming. In that setup, semi-supervised algorithm is very useful if only a small number of labeled examples are available. An example of semi-supervised learning algorithm is Co-Training. Here we employ Co-Training for phishing detection (see Section 5.3). Co-Training algorithm is presented below.

Co-Training

A classification task for phishing detection requires labeled phish examples and non-phish examples. While there are many data sources for obtaining general spam emails and good emails, there is very few labeled phishing email public corpus. As phishing emails and general spam emails share similar characteristics, human annotation result in incorrect labeling and hence the available corpus may not be perfect. Co-Training is an algorithm to solve this non-availability problem. The algorithm, proposed by Blum and Mitchell [18], for the problem of semi-supervised learning where there is both labeled and unlabeled examples. The goal of Co-Training is to enhance performance of learning algorithm when only a small set of labeled examples is available. The algorithm trains two classifiers separately on two sufficient and redundant views of the examples and lets the two classifiers label unlabeled examples for each other. The assumptions of the algorithm are that each view is conditionally independent given the class label and that each view is sufficient on its own for the purpose of classification. The algorithm works as follows: Given a set of labeled training instances ($L$) and a set of unlabeled instances ($U$), select $u$ instances randomly from $U$ to create a smaller pool $U'$. Iterate for $k$ iterations the following steps:
Step 1: Split each instance $x_i$ and build two views $x_1$ and $x_2$.

Step 2: Use the training set $L$ to build a classifier $h_1$ using $x_1$.

Step 3: Use the training set $L$ to build a classifier $h_2$ using $x_2$.

Step 4: Label $p$ positive and $n$ negative instances from $U'$ using the classifier $h_1$.

Step 5: Label $p$ positive and $n$ negative instances from $U'$ using the classifier $h_2$.

Step 6: Add labeled instances to the training set $L$.

Step 7: Select $2 \times (p + n)$ instances from unlabeled set $U$ and to add it pool $U'$.

The idea behind the Co-Training algorithm is that the classifier $h_1$ adds examples to the labeled set which are in turn used by the classifier $h_2$ in the next iteration and vice versa. This process should make classifiers $h_1$ and $h_2$ to agree with each other after several iterations. Blum and Mitchell [18] validated the co-training algorithm using 1051 web page data where $x_1$ consisted of words that appeared on the web page and $x_2$ consisted of words in all the hyper links that pointed to the web page. Nigam and Ghani [19] proposed a variant to the Co-Training called Co-EM algorithm. The Co-EM algorithm is not incremental in nature and it labels all unlabeled data at each iteration. Furthermore, only the data labeled by one classifier is used by the other classifier and vice versa. Co-Training was applied to the email domain by Kiritchenko and Matwin [20]. Authors used co-training to classify interesting versus uninteresting email. Chan et. al. [21] demonstrated Co-Training for spam classification on a corpus of 2883 emails. Wan [22] applied Co-Training to cross-lingual sentiment classification. Kumar and Daumé III [23] extended Co-Training to unsupervised spectral clustering algorithm.
where in clusters identified in one view is used to label data in other view so as to modify the graph structure.

We utilize Co-Training to evaluate the effectiveness of phishing detection on a large corpus of labeled and unlabeled data (see Section 5.3). The phishing detection methodology employs Co-Training to build classifiers on two views of the data, text view and hyperlink view, starting with few labeled samples and pool of unlabeled samples, and iteratively build a robust classifier for phishing detection.

3.2.4 Transduction

Transduction, a technique developed by Valdimir Vapnik [24], involves reasoning from observed training examples to test examples. Suppose there is a collection of points where most of them are unlabeled and few of them have labels, the inductive algorithm use labeled points to train a supervised learning algorithm and predicts the label of unlabeled point. This approach would result in poor performance if the labeled data is not large. In contrast, transduction considers all the points while performing the labeling task, by labeling the point to a cluster the point belongs. An example algorithm that employs transduction is Transductive Support Vector Machine. Face recognition using ARCF (see Section 4.3) utilizes both training and test data similar to Transduction.

3.3 Natural Language Processing

Natural language processing (NLP) is a computational technique for analyzing natural (human) language. It stems from the field of computational linguistics and artificial intelligence. There are six main types of NLP, namely, phonology - dealing with interpretation of speech sounds within and across words, morphology - dealing with
prefixes and suffixes, lexical – dealing with meaning of words, syntactic – dealing with grammatical structure in a sentence, semantic - dealing with disambiguation of words in multiple senses or context, and, discourse – dealing with text longer than a sentence.

Modern NLP methods rely heavily on machine learning to perform human like natural language processing. The part-of-speech tagging, which involves determining the correct part of speech of each word in a given sentence, employs machine learning technique to extract the correct part-of-speech for unseen word. It involves a training phase where sentences and words in a large corpus are correctly tagged. The corpus is analyzed and a model is learnt that consist of a set of rules to correctly predict the tag for a new unseen word in the testing phase. Some of the applications of NLP are listed below.

Automatic Summarization – The goal is to extract essence of the given text that provides a good summary.

Named Entity Recognition – This involves extracting proper names such as names of people, locations and organizations from a given body of text.

Translation – Automatically translate text from one language to another.

Optical Character Recognition – This involves extract text from an image.

Topic Discovery – Given a corpus of text, this involves discovering topics that are present in the corpus. Various other tasks using NLP include speech recognition, sentiment analysis, information retrieval and information extraction.

This research employs semantic analysis methods called Probabilistic Latent Semantic Analysis (PLSA) and Latent Dirichlet Allocation (LDA) for phishing topic
discovery and Conditional Random Field (CRF) for impersonated entity discovery. PLSA and LDA methods are discussed in Sections 3.3.1 and 3.3.2, respectively. CRF method is discussed in section 3.3.3. Application of these methods for phishing detection and impersonated entity discovery is presented in Chapter 5.

3.3.1 Probabilistic Latent Semantic Analysis

PLSA is a method for topic discovery proposed by Hoffman [25, 26]. The method is closely related to Latent Semantic Analysis (LSA). While LSA is based on the foundations of linear algebra that performs a Singular Value Decomposition (SVD) of co-occurrence tables, PLSA is a statistical methodology that defines a latent class model to perform probabilistic mixture decomposition. PLSA handles both synonyms, different words with similar meanings, and polysemy, words that mean differently in different context. PLSA has been applied in the field of information retrieval, natural language processing, machine learning and image processing.

PLSA model maps the high dimensional vector of words of a document to a lower dimension vector of topics. Suppose we have a collection of documents, \( d_i \) \( \{d_1, d_2, \ldots, d_I\}\), set of words that occur in those documents \( w_j \) \( \{w_1, w_2, \ldots, w_J\} \), the PLSA model associates a latent topic variable \( z_k \) \( \{z_1, z_2, \ldots, z_K\} \) with the occurrence of each word in a particular document. PLSA model assumes conditional independence. Thus, word and document are conditionally independent given topic. Thus, the PLSA model for the word-document co-occurrence can be expressed using the following join probability model:

\[
P(d_i, w_j) = P(d_i) \sum_{k=1}^{K} P(w_j|z_k) P(z_k|d_i)
\]
where, \(P(d_i)\) – is the probability that a word will be observed in a given document \(d_i\),
\(P(w_j|z_k)\) – is the probability of a particular word conditioned on latent topic variable \(z_k\),
and, \(P(z_k|d_i)\) – is the probability distribution of specific document over the latent variable space. The probability \(P(w_j|z_k)\) gives words that make up a given topic while the probability \(P(z_k|d_i)\) gives topics that a given document belong to. Unlike traditional cluster algorithm wherein a document may belong to just one cluster, PLSA gives the probabilities with which a given document may belong to one or more topics.

The model parameters \(P(w_j|z_k)\) and \(P(z_k|d_i)\) are estimated by maximizing the data log-likelihood using Expectation Maximization (EM) algorithm.

Step 1: The maximum likelihood formulation is given as follows:
\[
\ell = \sum_{d_i \in D} \sum_{w_j \in W} n(d_i, w_j) \log P(d_i, w_j)
\]

Step 2: By applying Bayes’ rule, E-Step of the EM algorithm is given by:
\[
P(z_k|d_i, w_j) = \frac{P(z_k)P(w_j | z_k)P(d_i | z_k)}{\sum_{t \in K} P(z_t)P(w_j | z_t)P(d_i | z_t)}.
\]

Step 3: The M-Step obtained by maximizing the expected data log likelihood is given by following expressions:
\[
P(w_j|z_k) = \frac{\sum_{i \in I} n(d_i, w_j)P(z_k | d_i, w_j)}{\sum_{m \in M} \sum_{i \in I} n(d_i, w_m)P(z_k | d_i, w_m)}
\]
\[
P(z_k|d_i) = \frac{\sum_{i \in I} n(d_i, w_j)P(z_k | d_i, w_j)}{n(d_i)}
\]
The model parameters are estimated by iteratively alternating the E-Step and the M-Step until desired termination criteria is satisfied. Stopping criteria may include no measurable difference in the log-likelihood between successive iterations or the maximum number of iterations.

In order for the PLSA model to generalize well on “new” (unseen) documents, Hoffman proposed a modified EM algorithm for PLSA called Tempered EM algorithm. TEM is closely based on deterministic annealing. In TEM, a control parameter $\beta$ is introduced in the E-step of the algorithm.

Step 1: The modified E-step is given as follows:

$$P(z_k | d_i, w_j) = \frac{P(z_k)[P(w_j | z_k)P(d_i | z_k)]^\beta}{\sum_{i \in K} P(z_i)[P(w_j | z_i)P(d_i | z_i)]^\beta}.$$ 

In the above expression, substituting $\beta$ with value of 1 yields E-step of standard EM algorithm. The main advantage of the TEM algorithm over the standard EM algorithm is that TEM avoids model over-fitting. The optimal value of $\beta$ is obtained by starting with a value of 1, evaluate the performance of EM on a held-out data set, decrease the value of $\beta$, and check if the performance improves. The iterative procedure is stopped when there is no measurable increase in performance.

Folding-In

When a new (unseen) document is given, to compute the probability distribution of topic(s) that new documents belong to, folding-in technique is employed. In PLSA, this is achieved by keeping the distribution of words that make up a topic ($P(w_j | z_k)$) fixed.
while the distribution of topics that new document belong to (P(z_k|d_{new})) is adapted at each M-step. The distribution P(w_j|z_k) is obtained during the training phase of PLSA.

Step 1: The E-Step of the EM algorithm for folding-in is given by:

$$P(z_k|d_{new}, w_j) = \frac{P(z_k)P(w_j|z_k)P(d_{new}|z_k)}{\sum_{i \in K} P(z_i)P(w_j|z_i)P(d_{new}|z_i)}.$$ 

and, for the M-Step, P(w_j|z_k) is obtained from the training phase.

Step 2: The distribution of topics to new document is given by the following expression:

$$P(z_k|d_{new}) = \frac{\sum_{i \in M} n(d_{new}, w_j)P(z_k|d_{new}, w_j)}{n(d_{new})}.$$ 

Folding-in technique yields probability distribution of new (unseen) documents belonging to one or more topics (P(z_k|d_{new})). Given a training set of labeled samples, belonging to one or more categories, category of the new unlabeled document is obtained using a similarity function. After obtaining probability estimates using fold-in, to categorize new documents to a specific category, similarity score between new documents and documents in the training set are computed. The category of the document in the training set that yields the highest similarity score is the category of the new document. Commonly used similarity function is the Euclidean distance function. However, Euclidean distance is not a good metric for computing similarity between two probability distributions. Hoffman derived the following Fisher-Kernel functions for the
generative statistical PLSA model. The kernel consists of two components. The Kernel function due to the contribution of topic probabilities is given by:

**Equation 13**

\[
K_1(d_i, d_n) = \sum_{k \in K} P(z_k \mid d_i)P(z_k \mid d_j)/P(z_k).
\]

The above kernel function computes the overlap between topics and thus captures words with similar meanings and words that belong to the same topic. The contribution due to word to topic probability distribution is given by the following kernel function.

**Equation 14**

\[
K_2(d_i, d_n) = \sum_j P(w_j \mid d_i)P(w_j \mid d_n) \sum_{k \in K} \frac{P(z_k \mid d_i, w_j)P(z_k \mid d_n, w_j)}{P(w_j \mid z_k)}.
\]

In the above kernel function K2, words with multiple meanings (polysemy) contribute to the similarity score. The PLSA method is applied to develop the multi-layered phishing email detection (see Section 5.3).

**3.3.2 Latent Dirichlet Allocation**

Latent Dirichlet Allocation (LDA) is a natural language processing method that discovers topics from a collection of documents. Documents are represented as random mixtures over latent topics and each topic is represented by a distribution over words. LDA, developed by Blei et al. [27], is built on the foundations of PLSA [25, 26]. Given parameters \( \alpha \) and \( \beta \), the LDA model is expressed as the joint probability distribution of a topic mixture \( \theta \), a set of \( N \) topics \( z \), a set of \( N \) words \( w \) using the following expression:

**Equation 15**

\[
p(\theta, z, w \mid \alpha, \beta) = p(\theta \mid \alpha) \prod_{n=1}^{N} p(z_n \mid \theta)p(w_n \mid z_n, \beta)
\]
The probability of a corpus D, computed from the product of marginal probabilities of individual documents, is expressed as follows:

**Equation 16**

\[
p(D | \alpha, \beta) = \prod_{d=1}^{M} \int_{\theta_d} \prod_{n=1}^{N_d} \text{SUM}_{z_{dn}} \left( p(z_{dn} | \theta_d) p(w_{dn} | x_{dn}, \beta) \right) d\theta_d.
\]

Several algorithms have been developed to solve LDA that requires estimation of the posterior probability distribution of hidden topic variables. It includes expectation-maximization algorithm [27], expectation-propagation algorithm [28] and collapsed Gibbs sampling [29]. The phishing website detection methodology (see Section 5.4) employs LDA to discover topics from a corpus of phishing and good website contents. It utilizes a collapsed Gibbs sampler algorithm for training and model inference. LDA topic distribution probabilities of phishing and good topics are used to build the phishing website classifier using the boosting algorithm, AdaBoost, which was described earlier in Section 3.2.1.

### 3.3.3 Conditional Random Field

Condition Random Field (CRF) is a sequence modeling method applied in this research for impersonated entity discovery (presented later in Section 5.5). Given a vector of input features \( x = \{x_1, x_2, \ldots, x_n\} \) and a vector of output variables \( y = \{y_1, y_2, \ldots, y_m\} \), a generative model estimates the joint probability distribution \( p(y, x) \) and a discriminative model estimates the conditional probability distribution \( p(y|x) \). The main difference between the two is that the discriminative model does not include a model of \( p(x) \). The Conditional Random Field (CRF) is a discriminative, uni-directed graphical model that models the probability distribution \( p(y|x) \). The nodes can be divided into two disjoint sets
$x$ and $y$, the observed and output variables, respectively. The CRF model can be expressed mathematically using the following equation [30]:

**Equation 17**

$$p(y | x) \propto \exp \left( \sum_j \lambda_j t_j(y_{i-1}, y_i, x, i) + \sum_k \mu_k s_k(y_i, x, i) \right)$$

where, $t_j(y_{i-1}, y_i, x, i)$ - transition feature function of the entire observation sequence and labels at positions $i$ and $i-1$ in the sequence, $s_k(y_i, x, i)$ - state feature function of the label at position $i$, and the observation sequence $\lambda_j, \mu_k$ - parameters estimated from training data. In order to simplify the above equation, the following two additional expressions are introduced:

**Equation 18**

$$s(y_i, x, i) = s(y_{i-1}, y_i, x, i)$$

**Equation 19**

$$F_j(y, x) = \sum_{i=1}^{n} f_j(y_{i-1}, y_i, x, i)$$

Thus, the probability of $y$ given observed variable $x$ can be written as follows:

**Equation 20**

$$p(y | x, \lambda) = \frac{1}{Z(x)} \exp \left( \sum_j \lambda_j F_j(y, x) \right).$$

$Z(x)$ is called the normalization factor. The CRF model parameters are estimated using log likelihood function, expressed using the following expression:

**Equation 21**

$$L(\lambda) = \sum_k \left[ \log \frac{1}{Z(x^{(k)})} + \sum_j \lambda_j F_j(y^{(k)}, x^{(k)}) \right].$$
The parameters cannot be determined analytically. Instead, parameters of CRF are obtained using an iterative procedure such as iterative scaling or gradient methods [30]. CRF is employed for impersonated entity discovery described later in Section 5.5.

3.4 Performance Evaluation

The performance of machine learning algorithms is evaluated experimentally instead of analytically. The evaluation method typically involves splitting the entire data set into distinct training set, validation set and a test set. The training set is used to train the system. The validation set is used to optimize parameters during training. The test set is used to evaluate the learned system. The larger the training set the better the learned model. The larger the test set, the more accurate the error estimates are. The performance of the learned model depends on type of learning algorithm employed, class distribution, misclassification cost, size of training data set and size of the test data set.

3.4.1 Training and Testing Methods

In hold out, also called splitting method, the entire data set is split into two disjoint sets. Every instance used for training is not used for testing and vice versa. The unseen test instances provide an unbiased estimate of system’s predictive accuracy. This method is employed when the data set is large. A commonly employed split is 2/3rd for training and 1/3rd for testing. Stratified sampling, a way of balancing the data, is used for small and unbalanced data sets. This method ensures each class is represented in approximately equal proportions in the training and test set. In cross-validation, the entire data set is partitioned to k folds of equal size. One fold is used for testing while training is done on remaining k-1 folds. The process is repeated for each of the k-folds. A special
form of k-fold cross validation is the leave-one-out cross validation where k is the size of the data set. This procedure is computationally expensive especially for large data set. The bootstrap sampling uses sampling with replacement to form the training set. We employ k-fold cross validation for performance evaluation of face recognition and phishing detection methodologies.

3.4.2 Performance Metrics

In a binary classification problem, where the question is to learn a way to classify unseen examples into one of two categories, positive categories and negative categories, True Positive (TP) means the actual and predicted categories are positive, True Negative (TN) means actual and predicted categories are negative, False Positives (FP) means the predicted should have been negative instead classified as positive, and False Negatives (FN) means predicted should have positive instead classified as negative.

Commonly used performance metrics in classification problems are accuracy, precision, recall, specificity and F measure. They are defined as follows. Accuracy is a measure of how accurate the learned system makes prediction on unseen test instances. Precision is defined as the proportion of true positives against all the positive results. Recall is the ratio of number of instances correctly classified to total number class instances. F-measure is the harmonic mean of precision and recall estimates.

Equation 22

\[
\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}
\]

Equation 23

\[
\text{Precision} = \frac{TP}{TP + FP}
\]
The quality of the PLSA model is evaluated using two measures of performance, namely, log-likelihood and perplexity. The training data set is split into a set for building the model (training data) and a set (held out) for validating the model using these performance measures.

**Log Likelihood:** The log likelihood on the training data set can be computed using the following expression.

\[
\text{Equation 27} \quad l = \sum_{d_i \in D} \sum_{w_j \in W} n(d_i, w_j) \log P(d_i, w_j)
\]

**Perplexity:** Perplexity, a measure of uncertainty in natural language models, gives a better assessment of how well the model generalizes on unseen (new) data. The lower the perplexity, the better the generalization and hence the classification. Perplexity for topic models PLSA and LDA is defined as follows:

\[
\text{Equation 28} \quad \text{Perplexity} = P = \exp \left[ \frac{-\sum_{h,j} n(d_h, w_j) \log P(w_j \mid d_h)}{\sum_{h,j} n(d_h, w_j)} \right]
\]
where, $n(d_h, w_j)$ is the number of times the word $w_j$ occurs in held out document $d_h$ and $P(w_j|d_h)$ is the probability that word $w_j$ occurs in document $d_h$. One can see that classification is proportional to the number of topics.

The above performance metrics are computed to compare the performance of our face recognition and phishing detection methods with state-of-the-art methods (see Chapters 4 and 5).

3.4.3 Receiver Operating Characteristic (ROC) and Area Under ROC Curve (AUC)

ROC curve is a plot of true positive rate versus false positive rate. Each point on the ROC curve represents different tradeoff between false positives and false negatives. The slope of the line tangent to curve is defined as the cost ratio. If the two ROC curves do not intersect, it implies that one method dominates the other method. The two-dimensional depiction of classifier performance in a ROC curve is reduced to single scalar value representing expected performance by computing the area under the ROC curve (AUC). The AUC measure has an important statistical property. The AUC of a classifier is equal to the probability that a classifier will rank a randomly chosen positive example higher than the randomly chosen negative example. Both ROC and AUC are computed to compare the performance of face recognition and phishing detection with state-of-the-art methodologies (see Chapters 4 and 5).
4 FACE RECOGNITION

One of the grand challenges for computational intelligence, in general, and computer vision, in particular, is to understand how people process and recognize each other’s face and to develop reliable face recognition systems. The face recognition challenge underlies biometrics, the science of authenticating people by measuring their physical or external appearance (and/or their behavioral or internal traits). In addition to security and surveillance, the ability to recognize living creatures has become also a critical enabling technology for a wide range of applications that includes defense, healthcare, human–computer interaction, image retrieval and data mining, industrial and personal robotics, and transportation. Two methodologies for reliable face recognition namely, (i) a novel adaptive and robust correlation filters (ARCF) and, (ii) a hybrid methodology that combines appearance based and anthropometric features, are presented here.

The ARCF methodology is based on recognition-by-parts strategy. ARCF provide information that involves both appearance and location. The cluster and strength of the ARCF correlation peaks indicate the confidence of the face authentication made, if any. The development of ARCF, motivated by MACE filters (see Section 3.1.4) and adaptive beam-forming from radar/sonar, is driven by Tikhonov regularization. The adaptive aspect of ARCF comes from their derivation using both training and test data, similar to
transduction, while the robust aspect benefits from the correlation peak optimization to
decrease their sensitivity to noise and distortions. Experiments are conducted illustrate
the feasibility and usefulness of the proposed ARCF approach vis-à-vis occlusion,
scrambled parts, disguise, illumination, and temporal and face expression variability.

The hybrid methodology combines appearance-based recognition (PCA or PCA +
LDA “Fisherfaces”) and holistic anthropometric-based recognition, with the latter
including Head (H) and Shoulder (S) in addition to Face (F) linear and non-linear
geometric measurements. The first processing stage considers feature extraction and
selection with PCA and PCA + LDA employing an energy based cut-off criteria to select
the eigenfaces basis, and the anthropometric features chosen using a correlation based
feature selection. Top ranked appearance-based and anthropometric-based features are
then optimally combined in a second stage using backpropagation or boosting.
Experiments for both occluded and disguised face images are carried out to evaluate
recognition performance, namely, i) stand alone “benchmark” PCA or PCA + LDA, ii)
boosting that employs feature-level fusion of PCA or PCA + LDA and holistic
anthropometric measurements, and iii) decision-level fusion that utilizes PCA or PCA +
LDA augmented by both holistic and face only anthropometric features.

The outline for the chapter is as follows. Section 4.1 motivates the need for
reliable face recognition. Section 4.2 reviews related research of face recognition
methodologies that address occlusion and disguise/masking. This includes methods such
as feed-forward cortical architectures, configural and holistic neural processing involved
with face recognition, recognition-by-parts computational strategies, correlation filters
and the role they play for recognition-by-parts, discriminative methods and boosting. Section 4.3 presents the novel adaptive and robust correlation filters (ARCF) methodology for face recognition. Anthropometric and appearance-based recognition is presented in Section 4.4. Methodologies developed here are summarized in Section 4.5.

4.1 Motivation

Denial and deception are characteristic of occlusion and disguise/masking, respectively, and affect biometric analysis. Biometrics cannot assume that personal signatures are complete and reliable. Occlusion and disguise are not necessarily deliberate. They can take place in crowded environments, e.g., CCTV, when only parts of faces are visible from time to time, or because of temporal changes, e.g., aging. Current face recognition engines are not built to withstand denial and deception, and their effectiveness as a consequence suffers. Deception is ubiquitous and diverse in nature. Essential for survival, it is one of the forces that drive natural selection. “Among all [these] evolutionary achievements, perhaps none are more important, more widely used, and more highly developed, than those characteristics which serve to elude, to attract, or to deceive the eye, and facilitate escape from enemies or the pursuit of prey” [31].

Deception, like camouflage, is most effective in cluttered environments when it becomes easier to hide, due to distracters and heavy cognitive load. Examples of phenomena with deceptive impact include bags under the eyes and wrinkles from aging, changes in appearance due to the use of cosmetics, medical condition (injuries and allergies), fatigue, hair style and facial hair. “Our face reflects the lifelong pull of gravity, which lengthens the jaws and deepens orbital bags. In addition people develop fat pads under
the eyes, shadows fall differently on the upper and lower eyelid” [32]. The working hypothesis for the (large) face recognition evaluations carried out so far has not been particularly concerned with the very possibility that subjects would seek to deny and/or foil their true biometric signatures. Most clients are legitimate and honest. They have nothing to hide, and have all the incentives to cooperate. The very purpose of biometrics, however, is to provide security from impostors and those seeking to breach security. It is quite obvious that such clients are well motivated to interfere with the proper acquisition of their biometric signatures, and will attempt to hide and/or alter the information needed for their authentication. Large scale face recognition evaluations, e.g., FRVT2002, FRGC, FRVT2006, still do not consider occlusion and disguise for testing purposes.

As occlusion and disguise usually affect only parts of the face, evidence accumulation through recognition-by-parts methods appear suitable for reliable face recognition. The ARCF methodology is developed to demonstrate recognition by parts strategy. In addition, this research develops a hybrid methodology that combines appearance (PCA or PCA + LDA) and holistic anthropometric features that include Head (H), Face (F), Neck (N), and Shoulder (S) linear and non-linear geometric measurements. The motivation and novelty of this hybrid approach comes from the realization that faces (and thus identities) are authenticated not in isolation (cropped and boxed) but rather as meaningful components of the head – face – neck - shoulders configuration. This is a straightforward extension of soft biometrics [33], which takes advantage of complementary information to facilitate and enhance face recognition. We next review
related research of face recognition methodologies that address occlusion and disguise/masking.

4.2 Background

Methods that address the occlusion and disguise problem are discussed here. Kohonen [34] used the error-correcting properties of orthogonal projections to show that linear auto-associative memory recall can withstand missing or noisy fragments for a small collection of faces encoded using only eight gay levels. Distributed associative memories (DAM), which are a generalization of associative memories and similar to holographic memories, can learn stimuli-response associations and display resilience to noise, partial occlusion, and (random or continuous) memory faults (“brain damage”) [35, 36]. Invariance to linear transformation is achieved using conformal mapping during pre-processing. Gross et al. [37] have used eigen-subspaces derived using a sliding window to define ID(entity) references as a sequence of signatures. A face of unknown identity is compared then with the stored reference sequences using Dynamic Space Warping (DSW), a variation on Dynamic Programming (DP) used for speech recognition. The image sizes used were small and the occlusions considered were limited in scope. The database available included a very large number, about 60 face images for each client, to accommodate parameter estimation for DSW. Real face recognition scenarios, however, have to handle large galleries of clients with very few photos per client. Everson and Sirovich [38] have proposed the Karhunen Loeve transform to provide an unbiased estimate for “gappy” data and fill them accordingly. The marred
faces were reconstructed and the gaps filled in a reasonable manner. The authors do not report any use of their method for actual biometric authentication.

Martinez [39] has suggested a (Gaussian or mixture of Gaussians) probabilistic approach that attempts to model the variation in image appearance due to errors in both face (and facial landmarks) localization and partial occlusion. To resolve the occlusion problem, each face is divided into \( k = 6 \) local but contiguous regions which are analyzed in isolation and their results aggregated. One major drawback of the method, which makes it impractical for real use, as explained by Martinez is due to the fact that “the ground-truth data (i.e., the correct localization of every feature on each face) is needed in order to estimate the error of a given localization algorithm. The problem is that the ground-truth data has to be obtained manually, which is a cost to be considered.” Even more challenging is the fact that ground truth is required for both training and test data. Matching expression variant faces can be facilitated by using motion estimation and dynamic cues [40]. The role that subset modeling plays for dealing with face occlusion is discussed by Martinez and Zhang [41]. Tan et al. [42] further the approach used by Martinez using Self-Organized-Feature-Maps (SOMF) instead of the mixture of Gaussians. Their method also requires manual annotation and needs to be told ahead of time about occlusions and their location. Matching expression variant faces can be facilitated by using motion estimation and dynamic cues [43].

Park et al. [44, 45] propose to handle occlusions, expression changes, and illumination variations using a set of simple lines that characterize the face structure [46] and the binary relations that hold among them. Some of the motivation behind the
proposed line edge map (LEM) and its corresponding mass structure comes from psychology. Psychological studies have shown that sketches, e.g., photofits, preserve most feature information and are less sensitive to illumination changes, or quoting the authors “face features are perceived in a holistic (or configural) manner with some kind of interactions between features by human beings.” The method proposed by Park et al. [44, 45] goes beyond LEM to include partial (flexible) attributed relational graph (ARG) matching between the face-graphs that include LEM for nodes and their relationships as edges. LEM distortions, e.g., position error, loss of line segments, and/or broken line segments, sensitivity to parameter setting, and complexity affect the robustness of the proposed method. There are similarities between elastic graph matching and LEM + ARG face graphs used. While the underlying representations are different, Gabor jets vs. LEM, respectively, both methods share the complexities associated with graph matching. The ARCF architecture described later on gets rid of the costs associated with graph matching and gains in efficiency. Yacoob and Davis [47] suggest to combine face recognition and hair analysis for improved authentication of occluded and disguised faces, e.g., faces with eyeglasses and neck/head cover.

Humans can detect and identify faces with little or no effort even if only partial views (due to occlusion) are available. This skill is quite robust, despite changes in the visual stimulus due to viewing conditions and expression. Partial faces are all what is sometimes available for training and/or testing, e.g., Face in a Crowd characteristic of CCTV. Martinez [39] reports different recognition rates for the left and right face images. Gutta and Wechsler [48], on a much larger data set, have shown, however, that the left
half, right half and the full face yield similar performance (about 95%) when matched against similar types of face images. The method that was used was based on the Ensembles of Radial Basis Functions (ERBF) network, whose design and implementation take advantage of asymmetric faces, which are examples of visual “hallucination.” Faces were further recognized from either their left or right half images when the face recognition engine is trained on full faces and tested on asymmetric faces constructed from either the left or right half augmented by their mirror image [48]. The ERBF implementation involves a simple and fully automated method using asymmetric faces (for either training or testing) and yields excellent results on a larger data set compared to the methods mentioned earlier.

Vijaya Kumar et al. [49] review correlation filters for face recognition and illustrate their advantages due “to working in the frequency domain (i.e., 2D Fourier transform of the images), e.g., shift-invariance, graceful degradation, and closed-form solutions.” Graceful degradation is the result of “the integrative nature of the matching operation” characteristic of correlation, which is similar in nature to distributed associative memories (DAM) (see above). Correlation measures the relative matching of two functions for different shifts. This is similar to convolution (filters) except that now the template is not reflected about the origin. Computationally this is convenient because convolution in one domain (spatial) corresponds to straightforward multiplication in the other domain (frequency). The shift-invariance property of such filters maps translations for test images to their correlation output being shifted by the same amount. This gets
away with the need to center the test images. Relative but similar shifts for the parts of an image then provide strong evidence for successful authentication.

Similar to elastic graph matching and “unlike in simple low-pass and high-pass filters, the phase of the correlation filters is very important for pattern matching.” The reason is that the phase encodes for location and relative displacements, and can carry weight with face authentication. The magnitude varies slowly with position, while the phase rotates with a rate set by the spatial frequency or the wave vector of the kernels. The use of phase provides, however, two advantages for face recognition when properly controlled [50]. “Firstly, phase information is required to discriminate between patterns with similar magnitudes, should they occur. Secondly, since phase varies so quickly with location, it provides a means for accurate jet (“feature”) localization in an image.” This is particularly important for face registration and normalization, in general, and eye detection, in particular.

The computational demands for correlation filters on large-scale face recognition problems, e.g., Face Recognition Grand Challenge (FRGC), however, are demanding as they require computing millions of frequency (Fourier) transforms (FT) and similarity metrics. Vijaya Kumar et al. [49] proposed the class-dependence feature analysis (CFA) to meet the heavy computational demands involved and show the utility of CFA on FRGC phase-II data. CFA builds one class vs. the remaining classes (MACE) filters and employs them as projection bases for test images in a fashion similar to eigenfaces. Similarity metrics are then simple inner products and decisions are made using winner-take-all (WTA). Due to non linear distortions in human’s face’s appearance, non-linear
mappings (for both the filter and test image) that increase their apparent dimensionality hold promise. The dimensionality is only apparent because the computation can still be carried out in the original image space using kernels that satisfy the Mercer’s theorem [24]. Kernel CFA (KFCA) [51] were derived and applied to FRGC 2.0 data/experiment 4/on the verification task, which includes a gallery of 466 subjects and approximately 16,000 images vs. a query (probe) set of 8000 facial images. The results reported for KFCA, 57% at 0.1% FAR, indicate that KFCA outperforms PCA (12% given as NIST benchmark), Gram–Schmidt Linear Discriminant Analysis (GSLDA) and CFA. Savvides et al. [52] follow a similar approach using kernel discrete cosine transform (KDCT) features and report 91.7% recognition performance.

Matching expression variant faces can be facilitated by using motion estimation and dynamic cues [40]. Levine and Yu [53] review correlation filters and describe their classification performance in scenarios that involve variations in facial expression, illumination conditions and head pose. Despite their lack of immunity to noise and excessive sensitivity to intra-class variations, the authors make a strong case that “correlation filters classifiers, a relatively unheralded model-based approach, have a greater robustness and accuracy [for face authentication] than traditional appearance-based methods (such as PCA).” The experimental results reported show that phase-only unconstrained MACE filter (POUMACE) yields the best choice for facial matching (and authentication vis-a-vis impostors). In particular POUMACE has achieved 100% accuracy on the CMU facial expression database and the Yale frontal face illumination database, and slightly less in the head pose experiments.
Serre et al. [54, 55] have recently proposed a quantitative model that “accounts for the circuits and computations of the feed-forward path of the ventral stream of visual cortex. This model is consistent with a general theory of visual processing that extends the hierarchical model of Hubel and Wiesel from primary to extra-striate visual areas and attempts to explain the first hundred milliseconds of visual processing [including immediate recognition and before eye movements, attention, and high-level processes can play a role].” Alternate layers of S(imple) and C(omplex) cells build “an increasingly complex and invariant object representation in a hierarchy of stages by progressively integrating convergent inputs from lower levels.” Gabor filters (or alternatively SIFT operators) at different positions and scales are applied first followed by increasingly complex features (in terms of spatial support and scale) that are built up by alternating layers of template matching (bell shaped tuning) and maximum pooling operations. The proposed feed-forward architecture assumes that recurrent paths characteristic of back-projections are inactive, and that learning takes place in an unsupervised fashion leading to a “generic dictionary of shape-components from V2 to IT, which provides an invariant representation to task-specific categorization circuits in higher brain areas.”

The feed-forward aspect is a rather restricted version of the latency and evidence accumulation concepts advanced by Thorpe et al. [56] and reiterated by the result reported by Sinha et al. [57]. Evidence accumulation involves a steady progression in the way the visual information is processed and analyzed. “This comes from bandwidth requirements and the need for an early and fast impression, categorization or recognition of the input. Much of the processing required to achieve such a phenomenal amount of
computation in such a short time must be based on essentially feed-forward mechanisms.” Asynchronous spike propagation and rank order (rather than rate) coding are some of the means proposed to explain the speed with which “neurons in the monkey temporal lobe can respond selectively to the presence of a face” [58]. The most strongly activated neurons or processing units fire first, greater impact is assigned to the spikes with shortest latency to stimulus onset, and the order and relative strength in which this takes place is the (temporal) code used for recognition. Such processing squares well with sparse coding driven by suspicious coincidences [59] and has been shown to “generalize well to novel views of the same face [for identification] and to be remarkably resistant to image noise and reduction in contrast” [60].

The hierarchical and invariant aspect of feed-forward architectures draws inspiration from the Neocognitron [61], while the generic dictionary, inspired by the universal basis driven by the statistics of natural scenes [62], is the result of adaptation. Correlation filters can realize yet another but relatively higher C layer for the feed-forward architecture. As an example, Torre et al. [63] have suggested using correlation filters for both handling the small sample size (SSS) problem and better generalization. Towards that end they propose to generate the subspace \( \text{SA} \) for some image \( A \) such that \( \text{SA} = \text{span} \{ A_i \} \) with \( A_i \) filtered versions of the raw/original image \( A \). The feed-forward architectures proposed so far are limited to the ventral (“what”) part. They omit the complementary dorsal (“where”) cortical path where spatial information is encoded. The beneficial role that location plays for object recognition is well known and it has been
recently discussed by Mutch and Lowe [64]. The adaptive and robust correlation filters (ARCF) proposed here handle both what and where components.

Yovel and Kanwisher [65] have shown, using fMRI studies of the Fusiform Face Area (FFA), that face perception is domain rather than process specific. Subjects had to discriminate among pairs of upright or inverted faces or houses stimuli that differed in either the spatial distance among parts (configuration) or the shape of the parts. The FFA showed a much higher response to faces than to houses, but no preference for the configuration task over the part task.” Canonical or configural [configurations] of face parts were found to trigger greater response vs. randomly rearranged parts within the face outline in the amygdala, superior temporal sulcus (STS), and FFA [66]. Face processing, however, is more than just configural. Face perception “engages a domain – specific system for processing both configural and part-based information about faces” [65]. This should accommodate viewpoint or pose changes, occlusion and/or disguise, and temporal changes.

Evidence for the holistic face space comes from “the detrimental effects of manipulations that disrupt the holistic structure of the face but leave individual features intact”[67], e.g., scrambling of the face, misaligning the lower and upper halves of the face, and inverted faces. Moscovitch et al. [68] have argued that only a vertical half is necessary to activate configural face processing and that holistic processing has access to enough information to fill in for missing parts. McKone et al. [67] have shown that holistic processing can operate in isolation from (local) feature-based identification. In particular, they have shown that holistic processing is called for during fine
discrimination tasks (on upright but not inverted faces) when the local cues for identity are unreliable, e.g., faces lacking distinguishable features, heavy (structural) noise due to illumination, mismatch of orientation between gallery and probe, expression, and make-up (disguise) used for deception. Configural and holistic processing are thus effective with occlusion ("denial") and disguise / masking ("deception").

Image variability and correspondence using precise alignment are major challenges for object recognition. The compositional and structural recognition-by-parts solution proposed by Fischler and Elschlager [69] to address such challenges includes a combined descriptive scheme and decision [embedded] metric. The parts or nodes characteristic of recognition-by-parts methods are referred to as components, landmarks or patches, and are held together or connected by linkages or strings. Recognition-by-parts makes face authentication easier because it does not seek for invariance. Instead, it handles variability using flexible geometric modeling to compensate for pose changes and limited occlusion and distortions. Characteristic of recognition-by-parts are the related methods of Dynamic Link Architecture (DLA) [70] and Elastic <Bunch> Graph Matching (E<B>GM) [71].

The dynamic link architecture (DLA) / elastic graph matching (EGM) is a minimum distance classifier with respect to (a) scale space representation; and (b) some non-rigid string geometry connecting landmarks across the face. The string geometry is flexible enough to tolerate small changes in appearance, e.g., facial expressions, and to provide for limited invariance. The landmarks, e.g., the pupils, the corners of the mouth, the tip of the nose et al., are coarsely represented using Gabor jets. Iterative matching
seeks the minimum for an energy function that compares the jets for amplitude 
preservation, on one side, and estimates the relative displacement of jets’ locations (using 
phase for accuracy), on the other side. An approximate solution, which decouples the 
above computation, is found in two stages. First it is rigid matching, e.g., template 
matching, which scores for jets’ compatibility using the local neighborhoods surrounding 
the jets’ location. The second stage stretches (in a non rigid fashion) the grid used for 
mapping the face using local perturbations in order to find ways to decrease the energy 
function. Minimizing the energy function is computationally expensive. The proposed 
recognition-by-parts solution using ARCF becomes computationally feasible and much 
easier to implement because both stages are performed at once to yield quality of match 
and layout.

Face recognition using elastic bunch graph matching (E<B>GM) also requires to 
find the landmarks and to place them in correspondence. To find the landmarks one now 
employs some general face representation that accounts for image variability due to age, 
gender and diversity, e.g., human eyes can be shaped differently. Wiscott et al. [71], 
aware that “it would be too expensive to cover each feature combination by a separate 
graph,” decided to “instead combine a representative set of individual model graphs into 
a stack – like structure, called a face bunch graph.” Each face model is now spanned by 
landmarks, represented using a bunch of jets, i.e., an exemplar-based representation, 
which are connected by springs or edges that take the average distance among them for 
value. “An eye bunch, for instance, may include jets from closed, open, female, and male 
eyes etc. to cover these local variations.” The minimization of the energy function
includes now also searching, independent of each other, for the best jets or local experts, among the bunch dedicated to each landmark. The analogue for the bunch is a part that is indexed using an exemplar-based representation, which would result when patches surrounding the facial landmarks are subject to correlation filters that account for human diversity and have their outputs clustered.

Component-based face recognition has been shown to outperform global / holistic methods [72, 73]. Neuropsychological evidence, however, suggests that “face recognition based on holistic information can occur in isolation from recognition based on local feature cue [i.e., parts]. [Furthermore], local features provide insufficient information to ensure the accurate discrimination of identity and, thus, configural processing of the face is also necessary” [67]. The corollary is that the additional use of the whole face and its half sides for components can benefit the face recognition-by-parts paradigm. Our experimental evidence shown later on suggests, however, that the mix of parts and holistic (full face) configuration helps only with faces that do not experience occlusion and/or disguise. For the latter case, the standard recognition-by-parts paradigm works best. Another important suggestion made is that the compositional structure consists of coarse coding of shape fragments (“parts”) and their retinotopy (“geometry”), or equivalently that the parts’ selective features when processed record for both shape (“what”) and location (“where”) [74]. Matching for both representation and location can be done using ARCF as is shown later on.

The motivation behind correlation filters came from the need for distortion-invariant optical pattern recognition. The seminal paper by Hester and Casasent [75] on
synthetic discriminant function (SDF) led to much productive research on correlation filters reviewed recently by Levine and Yu [53].

Some of the motivation for boosting goes back to Marvin Minsky who argued that “it is time to stop arguing over what is best [for decision-making] because that depends on our context and goal. Instead we should work at a higher level of organization and discover how to build [decision-level] managerial systems to exploit the different virtues and evade the different limitations of each of these ways of comparing things.” Similar concerns were expressed by Levin Kanal who made the obvious observation that “no single model exists for all pattern recognition problems and no single technique is applicable for all problems. Rather what we have in pattern recognition is a bag of tools and a bag of problems.” This corresponds to ensemble of methods, voting methods, and mixtures of experts, with boosting the preferred method to implement them.

The basic assumption behind boosting is that weak learners can be combined to learn any target concept with probability $1 - \eta$. Weak learners are usually built around simple features (or parts) as stump functions that classify at better than chance (with probability $1/2 + \eta$ for $\eta > 0$). Towards that end, AdaBoost [11] adaptively and progressively re-samples the data to focus learning with the relative weights of misclassified samples increased after each iteration. Boosting involves choosing $T$ effective features $h_t$ to serve as weak (learners) classifiers to construct separating hyper-planes. AdaBoost implements margin optimization with the margin viewed as a measure of confidence or predictive ability. The minimization using greedy optimization is similar to logistic regression [24]. AdaBoost converges to the posterior distribution of (the
labels) \( y \) conditioned on (data) \( x \), with the strong but greedy classifier \( H \) becoming in the limit the log-likelihood ratio test. The multi-class extensions for AdaBoost are AdaBoost.M1 and .M2 with the latter focused now on both difficult data samples to recognize and labels hard to discriminate. Examples of face recognition using boosting include Viola and Jones [76], Lu et al. [77], and Li and Wechsler [78]. The studies referenced above suggest locality of processing. This is reflected in both ARCF and hybrid methodology, which is presented next.

### 4.3 Adaptive and Robust Correlation Filters

The ARCF methodology is based on recognition-by-parts architecture. ARCF provide information that involves both appearance and location. The cluster and strength of the ARCF correlation peaks indicate the confidence of the face authentication made, if any. The development of ARCF, motivated by MACE filters and adaptive beam-forming from radar/sonar, is driven by Tikhonov regularization. The adaptive aspect of ARCF comes from their derivation using both training and test data, similar to transduction, while the robust aspect benefits from the correlation peak optimization to decrease their sensitivity to noise and distortions. The ARCF methodology is described next.

#### 4.3.1 Methodology

ARCF is motivated in part by the correlation filters (CF) (see Chapter 3). Correlation Filters are suitable for implementing recognition-by-parts using template matching. The strength of the correlation peak indicates how well the training and test images match, while the location of the peaks indicates the relative shift between the training and test images. Recognition-by-parts involves matching the corresponding parts
and their relative location. One maintains the relative locations of the parts during training and testing in order to check for their alignment. This is accomplished by using masks that expose only the relevant part(s) and zero out the rest of the face. Several examples used to illustrate the possible use of correlation filters for recognition-by-parts are presented next.

Three masks are applied to the training face to extract out the face parts corresponding to the right eye (RE), left eye (LE), and nose (N), with the area outside the mask zeroed out (see Figure 4). The masks consist of boxes that surround the training face parts as shown in the left image. The three face parts are used to design three match filters (MF) that are used for recognition-by-parts. In the first example, the test image (on the right) is from the same subject. The matching scores (correlation peaks) for the face components are high and the peak locations are aligned.

The next example (see Figure 5) illustrates the case of parts that match but miss the proper alignment. The training and test images come again from the same subject. The test image has been artificially cut at the middle and pulled apart so that the interocular distance has increased and the nose is now split. The MF shows a good match for the eye components, a poor match for the corresponding nose component, and the peak locations for the eyes components do not align. Authentication fails. The last example illustrates the case for different subjects (see Figure 6). The peaks from the MF are weak and misaligned, and authentication fails.
Figure 4 Correlation Peaks and Peak Locations for the Same Subject Using Match Filters
Figure 5 Correlation Peaks and Peak Locations for the Same Subject with Distorted Parts
Figure 6 Correlation Peaks and Peak Locations for Different Subjects
The correlation filters do not take advantage of the information provided by the test data, e.g., noise and distortions, in the design of the filter. Similar to beam-forming [79], correlation filters should be designed such that they adapt and automatically tune out the actual noise/distortion from test data without making arbitrary assumptions about the structure of the noise. This would result in an adaptive correlation filter whose output correlation surface has an optimally low average side-lobe level. The correlation peak, however, would still be sensitive to noise / distortion. To make the correlation peak robust to noise / distortion, one can employ an adjustable loading parameter that can be derived using an approach motivated by beam-forming or alternatively using Tikhonov regularization. The loading parameter, based on the magnitude of the match filter weight, provides then also for the robust filter sought after.

The rationale for the above two pronged optimization can be explained as follows. The overall optimization has to minimize the average side-lobe so that the correlation peak (vs. side-lobes) will stand out for matching to occur. The optimization, however, has also deleterious effects on the correlation peak if the (face) parts are slightly mismatched due to small additive noise and / or structural (face) distortions. To cope with noise, either additive or structural, one needs also to optimize the overall filter design. The optimization of the correlation peak, which corresponds to minimizing the side-lobes, can, however, interfere with the need to display robustness against noise. More robustness makes the average side-lobe become larger and the correlation peak stands out less. The relevant trade-off is that the average side-lobe level will not reach minimum
when one simultaneously seeks for robustness against noise. The adaptive aspect of ARCF refers to the use of test data in addition to training data in the derivation of the filter. The filter can be said to adapt or to adjust itself based on the test data presented to it in order to minimize the average output side-lobe level. The robustness aspect refers to the ability to prevent small mismatches from significantly reducing the strength of the correlation peak. Adaptation and robustness have thus to work together to maximize the peak to side-lobe ratio (PSR) for better detection. The adaptive and robust aspects are discussed next.

**Adaptiveness**

If the noise/distortion in the test data can be measured, then it can be minimized directly. This approach is used by both MVSDF (see Equation 9) and OTF (see Equation 11) when $Q_n$, the noise power spectrum or covariance, is known. When $Q_n$ is not known, it is assumed to be white. We take here a different approach. Motivated by adaptive beam-forming, we propose to learn the noise / changes observed in the test data and to automatically adjust the correlation filter in order to minimize its response. This is accomplished by minimizing the output correlation energy due to test data while maintaining a unit response to unit training data.

**Equation 29**

Minimize $h^H D_x h$

Subject to $S^H h = d$

where $S = [s_1 \ldots s_M]$ and $d = 1_M$

with $D_x$ is a diagonal matrix containing the power spectrum of the test exemplar. The (still non-robust) Adaptive Correlation Filter (ACF) solution $h = D_x^{-1} S (S^H D_x^{-1} S)^{-1} d$
is similar to the MACE filter, except that $D_s$ is now replaced by $D_x$. The use of test data $D_x$, in addition to training data $S$, in the design of the filter, is different from previous approaches to correlation filter design, and has proved beneficial. The filter tunes itself to the “noise” present in the test data in order to reject it. The output correlation surface has an optimally low side-lobe level, irrespective of the actual structure of the noise. This is different from MACE, which lacks an optimization criterion to reject the noise from test data. It is also different from MVSDF and OTF where the noise information $Q_n$ must be known or has to be assumed to be white even when the actual noise/distortion is not.

**Robustness**

A robust correlation filter should produce a stable correlation peak that changes very little even when there is a large change in the strength of the distortion / noise. To minimize the sensitivity of the correlation peak to the noise / distortion level, one has to minimize the rate of change of the squared correlation peak with respect to the strength of the noise / distortion that is present. Let the squared correlation peak be $p = E\{|hHx|^2\}$

**Equation 30**

$$p = E\{h^Hxx^Hh\} = E\{h^H(s + n)(s + n)^Hh\} = E\{h^H(ss^H + sn^H + ns^H + nn^H)h\}$$

$$= h^Hss^Hh + h^HE\{sn^H + ns^H + nn^H\}h = h^Hss^Hh + h^HQh = h^Hss^Hh + \xi h^HNh$$

where the covariance $N$ is normalized so that the average of the diagonal elements is 1, and $\xi$ is the strength parameter. One seeks to minimize $dp/d\xi = h^HNh$. When the noise/distortion is not known, it is typically assumed to be white, $N = I$. The ARCF formulation then becomes:

**Equation 31**

Minimize the output correlation energy $h^HD_s h$
Subject to unit response to training signal $S^H h = d$

Subject to sensitivity constraint $h^H I h \leq \alpha$

The solution is: $h = (D_x + \varepsilon I)^{-1} S [ S^H (D_x + \varepsilon I)^{-1} S ]^{-1} d$ with $\varepsilon$ satisfying the constraint $h^H I h \leq \alpha$. The solution for $\varepsilon = 0$ is $h = D_x^{-1} S [ S^H D_x^{-1} S ]^{-1} d$.

It has the same form as the MACE filter, which is also sensitive to noise and distortion. The solution $h = S [ S^H S ]^{-1} d$ is found when $\varepsilon = \infty$. This is the same as the SDF filter and the correlation peak displays maximum robustness to white noise. The magnitude of the SDF weight is the smallest among the adaptive correlation filters with white noise robustness. Thus one choose $\varepsilon$ that satisfies the constraint $h^H h \leq k |h_{SDF}|^2$ with $k \geq 1$. We show below how different correlation filters compare in matching the left eye component for both representation and location. First we show the effects of aging (see Figure 7) and then the combined effects of aging and Gaussian noise (see Figure 8).

Note that the training image (08/93) and test images used to assess performance on aging (05/96) and aging (05/96) + noise N(0,0.1) for Figures 7 and 8, respectively are shown on top of Figure 7. Matching the left eye component against the whole face using various correlation filters is shown in Figure 7. The true peak is at the center of the horizontal axis. Note that the MF has the strongest true peak, but it also has significant false peaks. The MACE correlation peak is sensitive to distortion and is barely visible. OTF has a good true peak but it has also an equally strong false peak. Of the four correlation filters, only ARCF shows the largest separation between the true peak and the much weaker false peak, and has the lowest average side-lobe level. One can thus see that ARCF
outscores MF, MACE, and OTF in terms of discriminating between the true peak corresponding to the left eye and the false peak caused by the right eye. In addition, one notes that ARCF displays the lowest average side-lobe, which indicates its robustness to noise. The advantage of ARCF over the competing correlation filters becomes even more compelling when noise is added. The false peak for OTF shows now as the strongest (see Figure 8). The architecture of ARCF is presented in the next section.
Figure 7 Comparison of Correlation Filters
Figure 8 Effects of Both Temporal Change and Additive White Noise on MF, MACE, OTF, and ARCF
4.3.2 Architecture

The feed-forward ARCF architecture is characteristic of recognition-by-parts strategies (see Figure 9). The face parts for an enrolled client and their counterparts drawn from test data are combined component wise to build the corresponding ARCF filters. The outputs from ARCF are combined using Linear Discriminant Analysis (LDA) to learn optimal (separation) boundary hyper-planes. The ARCF outputs are then projected on the LDA axes found to find the authentication score. ROC at FAR = 1% using scores from both authentic and impostors claims determines in an a-priori fashion the optimal decision threshold used on future authentication claims. In particular, we found that for the purpose of generalization it is feasible to learn the optimal LDA axes and decision thresholds from one population, e.g., FERET [80], and use them on a different population, e.g., AR [81].
Preprocessing

Faces are rotated and scaled for the eye centers to align. A common full – face mask is applied to the image to extract the full face. The face is normalized by its mean and standard deviation to have zero mean and unit variance. A mean face is computed
from the whole population available for training. The final preprocessed face is the normalized face less the mean face.

**Face Parts**

Recognition-by-parts requires defining and modeling the face parts involved. A single but complete training face image yields multiple face parts by applying different masks to expose the face components used. Our architecture employs now seven face parts that correspond to the left eye, right eye, nose, mouth, left half–face, right half–face, and the full face without the hair. The relative location of these parts is retained such that their overlay reconstructs the original face. When the face parts are perfectly correlated with the original face the components come from, the correlation peaks align at a single point, i.e., the center of the coordinate system for the correlation surface. The strength of the correlation peaks indicates how well the trained face components match those found in the test image. Tight clustering of the locations for these correlation peaks indicates that the relative configuration of the face components found on the test image matches that of the training (gallery) face. Tight clustering of strong correlation peaks accounts thus for matching both appearance and relative (layout) locations.

**ARCF Filter Bank**

The face model consists of a collection of ARCF filters, one for each face part. Each ARCF filter corresponds to one of the face parts and is derived using both the training / enrolled face and the corresponding part from the test face. Multiple training
(from the same client) and/or test faces are allowed. The output of the ARCF filter bank is a vector containing the correlation peak strengths and their distances from the origin. The vector consists of 14 components corresponding to the 7 correlation peaks’ strength and 7 distances. The distance information is used to enforce location matching and eliminate false peaks. A peak whose distance exceeds the (empirically validated) threshold is considered a false peak and has its peak strength zeroed out. By having a bank of ARCF filters where each filter is trained on a single face component, instead of a single ARCF filter trained on all seven face components, the intermediate outputs for each face part can be monitored and analyzed before they are combined. The intermediate outputs can be further combined in a non-linear rather than linear fashion to exploit face symmetry as described below.

**Decision Stage**

Face symmetry is exploited using non-linear processing of the individual correlation peaks. For symmetric face components such as the left and right eyes, or the left and right half faces, we use the dominant correlation peak found. We compute three similarity scores for full – face (F), half – faces (HF), and face parts (P). F is the peak strength for the full – face, HF is the dominant peak strength among the left “H” and right “h” half – faces. P is a linear combination of [max (left eye “E”, right eye “e”), nose “N”, mouth “M”] using the weight w derived using LDA on FERET.

One finds w as the optimal projection that best separates the authentic class from the impostor class. w is normalized so the relative weights of the test components are
unchanged and any missing component (peak = 0) does not reduce P. For example, if \( \mathbf{w} = [0.57, 0.3, 0.13] \) and the eyes are missing, then the normalized \( \mathbf{w} = [0, 0.3, 0.13] / 0.43 \).

The thresholds for the three similarity scores \( [F, HF, P] \) were determined from ROC using FERET (training database) at FAR = 1\% to be \( [0.16, 0.2, 0.26] \) and used on other databases including AR. Authentication succeeds when any one of the three similarity scores is above its corresponding threshold. We report next the results from two sets of experiments conducted using FERET, AR and Essex face databases.

### 4.3.3 Experiments

The experimental results presented in this section show the feasibility and robustness of the recognition-by-parts architecture built around ARCF. The robustness is vis-à-vis disguise, scrambling the face parts (wrong configuration), occlusion, varying illumination, and temporal changes including varying facial expressions. Data sets and results are reported next.

**Data Sets**

Two sets of experiments were conducted, the first one using FERET and AR database and the second one using AR and Essex databases.

**FERET**: FERET database was a collaborative effort between Wechsler and Philips [80]. Images were collected in a semi-controlled environment. The data was collected in 15 sessions between August 1993 and July 1996. The database contains 14126 images of 1199 individuals.

**AR**: AR database was collected by Martinez and Benavente [81]. The database contains 4000 face images corresponding to 126 individuals (70 men and 56 women).
These images were taken under strictly controlled conditions. Each individual participated in two sessions, separated by two weeks.

**Essex**: Essex database was collected by the University of Essex, UK [82]. The database contains face images of 395 individuals (male and female), 20 images per individual. The database contains images of people of various ethnicity, their age ranging from 18-20.

**Results – Part I**

The face images used for Part I were selected from FERET and AR. The three reported similarity scores are F (Full Face), HF (maximum for half faces), and P (maximum for the eyes, nose, and the mouth). To facilitate interpretation, the corresponding thresholds for FAR = 1\% are subtracted so the threshold is exceeded for positive numbers only. If any of the three similarity scores exceeds its corresponding threshold, authentication succeeds (Accept). If all three similarity scores are below their corresponding threshold, authentication fails (Reject).

**Disguise**

The face images used for training come from [http://makeoversolutions.com](http://makeoversolutions.com), while the test images are obtained from the training ones by using a different hairstyle and adding sunglasses (see Figure 10). For both test images the parts’ similarity score $P$ is the strongest among the three similarity scores and it exceeds the threshold. Authentication succeeds to uncover the identity behind the disguise.
Scrambling of Face Parts

The face images used in this experiment (see Figure 11) come from both the FERET and AR databases. The right eye of the test image is moved up and toward the right. Scrambling of the face parts is indicated by the position of the right eye away from the center. Verification succeeds, however, since the left half of the face matches well. In the second test, the client is behind another person. This is also a case of occlusion but the left eye and left half face are detected and in proper alignment. The score based on matching the full face is negative (below the corresponding threshold) indicating that verification based on the full face has failed. Verification, however, still succeeds because the good match on the left half face “H” results in a good HF score and the good match on the left eye “E” also yields a good P score.

Occlusion

The faces, both for training and testing, come from the AR database (see Figure 12), but the acceptance thresholds and optimal LDA projection \( \omega \) were derived using FERET. The strong correlation peaks for the parts that are not occluded are aligned such that the parts’ similarity score \( P \) is the strongest among the three similarity scores and it exceeds the threshold. Authentication succeeds to uncover the identity behind the occlusion. The weak correlation peaks for the full face and eyes (see Test1 with sunglasses), and mouth (see Test2 with scarf) cannot prevent the ARCF decision stage from locking on the correct authentication in both cases. This experiment also shows that
holistic components, i.e., the whole face here, do not help with recognition of occluded faces.

Varying illumination

The face images used for training and testing come from AR (see Figure 13). The test images are different from the training ones in terms of varying illumination. The correlation peaks for the face seen below are strong and aligned, the similarity scores exceed the corresponding thresholds, and authentication succeeds.

Temporal change and varying face expression

The face images used for training and testing come from FERET (see Figure 14). The test images were acquired two years later compared to the training ones and face expression varies. The correlation peaks for the face seen below are strong and aligned, the similarity scores exceed the corresponding thresholds, and authentication succeeds.
Figure 10 ARCF Results for Disguise
Figure 11 ARCF Results for Scrambling of Face Parts
Figure 12 ARCF Results for Occlusion
Figure 13 ARCF Results for Varying Illumination
Figure 14 ARCF Results for Temporal Change and Varying Face Expression
Results – Part II

Recognition-by-parts using ARCF was illustrated in the previous section for single test images vis-à-vis occlusion, disguise, varying illumination, temporal changes, and wrong assembly of parts. ARCF performance is compared in this section against traditional face recognition algorithms such as PCA and PCA + LDA vis-à-vis disguise. Large scale face recognition evaluations, e.g., FRVT2002, FRGC, FRVT2006, do not consider disguise for testing purposes. Our own evaluation study described here shows that the performance displayed by well known face recognition methods, e.g., PCA and PCA + LDA (“Fisherfaces”), deteriorates significantly as a result of disguise. The data (http://cobweb.ecn.purdue.edu/~aleix/aleix_face_DB.html) used comes from AR for learning the dimensions of the face space, and from the University of Essex for building galleries and probe sets that consist of “clean” and/or “disguised” face images. The disguised (male) probes are generated by adding beards (using Adobe Photoshop) to an existing clean gallery of face images. University of Essex does not allow the publication of images from their database. Different databases are used for learning the face space, i.e., AR, and for training and testing, i.e., Essex. Such a protocol distinguishes learning the face space and training for gathering gallery signatures as two different processes. The AR face database has over 4,000 color images (from 126 subjects) with varying illumination, facial expressions and occlusions. The face recognition database (http://cswww.essex.ac.uk/mv/allfaces/) from the University of Essex has 7,900 images (from 395 subjects), which vary in terms of facial expressions, race, gender, lighting conditions, and facial hair. In addition, some of the individuals had glasses.
The Face Identification and Evaluation Software [83], developed by Colorado State University (CSU) (http://www.cs.colostate.edu/evalfacerec/index.html), is used for evaluation purposes on PGM normalized face images using manual annotation for eye coordinates. Photoshop was used to extract the seven parts from the original full face images to accommodate ARCF needs. To meet CSU requirements the raw images from the AR database and JPEG images from the Essex database are converted to PGM format using the ImageMagick library (http://www.imagemagick.org/script/index.php). The eye coordinates were manually obtained using MATLAB [84] getpts function. All images were resized to 96 x 72 pixels in size. Finally, a five-step normalization using CSU software was carried out for (i) converting 256 gray levels into floating points; (ii) geometric normalization that lines up the chosen eye coordinates; (iii) cropping the image using an elliptical mask; (iv) equalizing the histogram of unmasked part of the image; and (v) normalizing the pixel values to mean zero and variance of one.

Face recognition experiments were first run using a gallery that includes 50 (male) subjects from the Essex facial database who display a clean face and a neutral expression. The two probe data sets used correspond to the same 50 subjects. The first probe set includes original “clean” face images, possibly with different (not necessarily neutral) facial expressions. The second probe set (from Essex) includes disguised face images generated as explained above. The PCA basis eigenvectors are computed using 200 images from the AR. The final base retained (for building personal signatures) is spanned using a reduced 80 dimensional PCA subspace. Euclidean distance and nearest neighbor classifier are used for authentication and matching. Similar considerations are
used for PCA + LDA. The results displayed in Figure 15 show that disguise leads to a significant deterioration in performance for both PCA and PCA + LDA.

![ROC Curve](image.png)

**Figure 15 ARCF Results - ROC Curves for PCA and PCA+LDA Disguised Faces**

Comparative ARCF performance vis-à-vis PCA and PCA + LDA is reported next on images (with / without disguise) from the University of Essex database. Results for four different (training, testing) scenarios using PCA, combined PCA + LDA and ARCF for across two sets of experiments are tabulated (Table 1 and Table 2). The difference between the two tables is the AR data used for learning the face space, i.e., the base. The experiments tabulated by the two tables use “clean” images, and “clean” mages and
images with “occlusions” (sunglasses and scarf), respectively. The four scenarios. A – D, tabulated by both tables, are the Cartesian product of training (on clean or disguised) and testing on (clean or disguised) data. The tables record recognition rates for false acceptance rates (FAR) varying from 0.05 to 0.4. As can be seen from Tables 1 and 2, for the scenario where clean face images were used for both training and testing (1-A), the recognition rates varied from 0.75 to 0.96. The performance of PCA was better than combined PCA + LDA and ARCF for (table, scenario) combinations 1-A and 2-A. For all the other combinations (1-B, 1-C, 1-D, 2-B, 2-C, and 2-D), the performance of PCA was better than combined PCA + LDA. ARCF outperforms, however, both PCA and combined PCA + LDA for all the remaining combinations with recognition (hit) rates as high as 0.94 at a FAR of 0.4. ROC curves showing the superiority of ARCF against PCA and PCA + LDA, for scenarios with disguised images, are shown in Figure 16 and Figure 17.

Table 1 ARCF Results (1) - Recognition (Hit) Rates for PCA, PCA+LDA and ARCF

<table>
<thead>
<tr>
<th>FAR</th>
<th>Recognition (hit) rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-A</td>
<td>train = clean (neutral expression), test = clean (slight smile)</td>
</tr>
<tr>
<td>1-B</td>
<td>train = clean, test = disguised</td>
</tr>
<tr>
<td>1-C</td>
<td>train = disguised, test = clean</td>
</tr>
<tr>
<td>1-D</td>
<td>train = disguised, test = disguised</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PCA + LDA</th>
<th>PCA</th>
<th>ARCF</th>
<th>PCA + LDA</th>
<th>PCA</th>
<th>ARCF</th>
<th>PCA + LDA</th>
<th>PCA</th>
<th>ARCF</th>
<th>PCA + LDA</th>
<th>PCA</th>
<th>ARCF</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>0.75</td>
<td>0.76</td>
<td>0.75</td>
<td>0.2</td>
<td>0.3</td>
<td>0.65</td>
<td>0.2</td>
<td>0.25</td>
<td>0.65</td>
<td>0.2</td>
<td>0.62</td>
</tr>
<tr>
<td>0.15</td>
<td>0.81</td>
<td>0.81</td>
<td>0.81</td>
<td>0.26</td>
<td>0.35</td>
<td>0.88</td>
<td>0.30</td>
<td>0.30</td>
<td>0.88</td>
<td>0.30</td>
<td>0.88</td>
</tr>
<tr>
<td>0.25</td>
<td>0.90</td>
<td>0.90</td>
<td>0.90</td>
<td>0.34</td>
<td>0.46</td>
<td>0.97</td>
<td>0.48</td>
<td>0.48</td>
<td>0.97</td>
<td>0.48</td>
<td>0.97</td>
</tr>
<tr>
<td>0.35</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>0.42</td>
<td>0.54</td>
<td>1.01</td>
<td>0.56</td>
<td>0.56</td>
<td>1.01</td>
<td>0.56</td>
<td>1.01</td>
</tr>
<tr>
<td>0.45</td>
<td>1.05</td>
<td>1.05</td>
<td>1.05</td>
<td>0.50</td>
<td>0.61</td>
<td>1.05</td>
<td>0.61</td>
<td>0.61</td>
<td>1.05</td>
<td>0.61</td>
<td>1.05</td>
</tr>
</tbody>
</table>

Notes: clean images from AR database.
Train and test = clean and disguised images using FERET database.
Experimental evidence (Part-I and Part-II) shows the feasibility and reliability of ARCF vis-a-vis occlusion, disguise, and illumination, expression, and temporal variability. We next present the hybrid anthropometric and appearance-based face recognition methodology.

Table 2 ARCF Results (2) - Recognition (Hit) Rates for PCA, PCA+LDA, and ARCF

<table>
<thead>
<tr>
<th>Pair</th>
<th>2-A (train = clean, test = clean)</th>
<th>2-B (train = clean, test = disguised)</th>
<th>2-C (train = disguised, test = clean)</th>
<th>2-D (train = disguised, test = disguised)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PCA-LDA</td>
<td>PCA</td>
<td>ARCF</td>
<td>PCA-LDA</td>
</tr>
<tr>
<td>0.85</td>
<td>0.71</td>
<td>0.76</td>
<td>0.2</td>
<td>0.3</td>
</tr>
<tr>
<td>0.87</td>
<td>0.80</td>
<td>0.80</td>
<td>0.13</td>
<td>0.56</td>
</tr>
<tr>
<td>0.88</td>
<td>0.88</td>
<td>0.47</td>
<td>0.13</td>
<td>0.56</td>
</tr>
<tr>
<td>0.89</td>
<td>0.89</td>
<td>0.48</td>
<td>0.13</td>
<td>0.56</td>
</tr>
<tr>
<td>0.90</td>
<td>0.90</td>
<td>0.48</td>
<td>0.13</td>
<td>0.56</td>
</tr>
<tr>
<td>0.91</td>
<td>0.91</td>
<td>0.48</td>
<td>0.13</td>
<td>0.56</td>
</tr>
<tr>
<td>0.92</td>
<td>0.92</td>
<td>0.48</td>
<td>0.13</td>
<td>0.56</td>
</tr>
<tr>
<td>0.93</td>
<td>0.93</td>
<td>0.48</td>
<td>0.13</td>
<td>0.56</td>
</tr>
<tr>
<td>0.94</td>
<td>0.94</td>
<td>0.48</td>
<td>0.13</td>
<td>0.56</td>
</tr>
</tbody>
</table>

Notes: clean images and images with occlusions (goggles, scarf) from AR database.
Train and test = clean and disguised images using face database.
Figure 16 ARCF Results - ROC Curves (Basis-Clean, Training-Clean/Disguised, Testing-Clean/Disguised)
4.4 Anthropometric and Appearance-Based Recognition

The methodology combines appearance-based recognition (PCA or PCA + LDA “Fisherfaces”) and holistic anthropometric-based recognition, with the latter including Head (H) and Shoulder (S) in addition to Face (F) linear and non-linear geometric measurements. This hybrid methodology is described next.

4.4.1 Methodology

The recognition methodology proposed here combines two representation-based methods: appearance-based and anthropometric-based. For appearance-based representation, we employ two types of face spaces: PCA (eigenfaces) and PCA + LDA
(Fisherfaces) including dimensionality reduction using energy-cutoff criteria. The training (enrolled) images are pre-processed using histogram normalization, geometric normalization using the eye coordinates, and cropping the face images using an elliptical mask. The Euclidean distance computes the similarity between gallery and query images for the purpose of authentication. Anthropometric-based representation includes the extraction of geometric features and their iterative ranking and selection for the purpose of dimensionality reduction. The novelty regarding anthropometric-based representation here is the use of geometric measurements that include head and shoulders (below the neck) in addition to the face. Two types of data fusion for the purpose of human authentication operate on the representations derived, namely, feature-level fusion and decision-level fusion. The feature-level fusion methodology considers eigen coefficients and holistic anthropometric measurements as “weak learners” (stump) features. It employs boosting to select the most significant features for their assembly into strong classifiers. The nearest neighbor classifier scores the relative merits of weak learners for boosting. For decision-level fusion method, the similarity Euclidean scores obtained using both appearance-based and anthropometric-based representations are fed to connectionist (“neural networks”) training to estimate their relative combination weights.

**Appearance-based Features**

Eigen-faces are derived using Principal Component Analysis (PCA). They map the original high-dimensional raw face space into a lower uncorrelated dimensional face space. Fisherfaces are derived using a combination of PCA (for dimensionality reduction) and Linear Discriminant Analysis (LDA) for better class separation.
PCA + LDA representations have been widely employed to show top face authentication performance when occlusion and disguise are not present. The objective here is to assess the performance of such representations using degraded face images due to occlusion. The face identification evaluation system developed by Colorado State University (CSU) [http://www.cs.colostate.edu/evalfacerec/algorithms5.html](http://www.cs.colostate.edu/evalfacerec/algorithms5.html) [83] is used to assay the performance for both PCA and PCA + LDA representations. We follow CSU recommendations and use 60% of eigenfaces coefficients for recognition. For 200 images this amounts to a 120-D (face space) basis. As an alternative we also use 18 eigenfaces corresponding to the 90% energy cut-off.

**Anthropometric-based Features**

Previous research using anthropometric-based representations had access only to face features and evaluated their utility for authentication on “clean” images without occlusion or disguise. The purpose here is to assess the utility of anthropometric-based geometrical features on face images adversely affected by occlusion and/or disguise. Similar to soft biometrics, we extract traditional head and face features measurements, together with novel features below the face, which correspond to neck and shoulders. The extraction process is semi-automatic, with manual location of facial landmarks (using MATLAB [84] `getpts`) and automatic derivation for the feature values. The resulting 19 holistic (head – face – neck – shoulders) anthropometric features extracted are shown in Figure 18 and described in Table 3. The extracted features include horizontal and vertical distances, linear and non-linear (curves) measurements, and head, face, neck and shoulder measurements. The MATLAB’s `getpts` function is used to capture coordinates
and the Euclidean distance is then computed between different pairs of points. The anthropometric feature set also includes several non-linear measurements, namely, curve (arc) lengths. To measure the length of a curve, MATLAB’s spline toolbox is used to fit a smooth spline curve and the resulting curve length is then derived. The feature set includes curves related to the eyes, mouth, shoulder, neck, etc. The linear measurements include inter-eye distance, inter-shoulder distance, mid point of mouth to mid point of nose, etc.
Figure 18 Holistic Anthropometric Features
Anthropometric Feature Selection

Feature selection is the process of identifying and removing irrelevant and / or redundant attributes. This process results in dimensionality reduction, which helps to improve the recognition rate and minimizes computational resources during both training and testing. There are two main categories of feature selection, filter and wrapper based methods. The filter based approach employs the characteristics of the data rather than
those of the learning algorithm. The wrapper-based method finds features that work best with a particular learning algorithm. Filter based selection methods are generally faster and more efficient than wrapper-based methods. The criteria for features selection involve a ranking scheme using measures such as entropy, information gain, and Fisher score. The filter-based feature selection approach is conceptually driven by the mutual information between data and its label. The specific method used here to implement the filter-based approach is the correlation-based feature selection (CFS) [85]. The CFS algorithm is based on a heuristic that “good feature subsets contain features highly correlated with the class, yet uncorrelated with each other.” The algorithm computes the following heuristic fitness to select a quasi-optimal subset of features

**Equation 32**

\[ MERIT(FS) = \frac{N \times \text{corr}(F,C)}{\sqrt{N + N(N-1) \times \text{corr}(F,F)}} \]

with \( F \) the feature, \( C \) the class, \( N \) the total number of features, \( \text{corr}(F,C) \) the average of feature-class correlation, and the average of feature-feature correlation. The forward selection search algorithm is used to search the feature space. The normalized fitness ranges from 0 to 1. In order to perform feature selection, 19 anthropometric features (listed in Table 3) are extracted from the AR image face database [81] and the Essex DB [82] (http://cswww.essex.ac.uk/mv/allfaces/index.html). Anthropometric features are extracted from 90 clean images that included head (H), face (F), Neck (N) and shoulder (S). WEKA [86] (http://www.cs.waikato.ac.nz/ml/weka/) machine learning software is used to perform feature ranking and selection. The CFS algorithm in WEKA software is run on the data set starting with an empty feature subset. The algorithm employs the
forward search selection algorithm to generate the top ranked feature subset. The ranking of individual features after running the feature selection algorithm is listed in the third column of Table 3. The ‘top 13’ features selected using the feature selection algorithm are shown shaded, the darker shade representing features below the chin, and the lighter shade representing features that belong to the head and face. The merit of the best feature subset obtained (upon convergence) is 0.83. Four of the features selected (about a third of the features selected) include shoulder and neck measurements. Features that can not be localized due to occlusion and/or disguise have their weight proportionally redistributed among the features that can be detected during human authentication. We present next the architecture of the developed methodology.

4.4.2 Architecture

Once holistic appearance-based and anthropometric-based representations have been derived and their dimensionality reduced, matching and decision-making can take place. Towards that end, similarity distances and decision-making methods are proposed. Feature-level fusion using boosting and decision-level fusion using backpropagation are the specific decision-making methods proposed and they are described next.

*Feature-Level Fusion Using Boosting*

Boosting employs Adaboost [11], an iterative and adaptive classification algorithm that aggregates a sequence of weak (stump) classifiers by updating the importance for the data points according to the errors encountered in previous iterations. Here we employ an Adaboost version developed by Tieu and Viola [87] implemented
using WEKA [86]. The output for the algorithm is a set of $T$ features / weak classifiers after $T$ rounds of boosting. We employ two sets of features for the boosting algorithm, i) anthropometric features, and ii) eigen coefficients, obtained from the PCA algorithm using an energy cut-off criteria. The input to boosting consists of holistic anthropometric features (as described below) and 18 eigen (PCA) coefficients (for the 90% energy cut-off criteria). The top ranked 25 features selected by the boosting algorithm are used for human authentication. Since face recognition is a multi-class recognition identification problem, a majority voting strategy combines all the pair-wise classification results.

**Ranking of Features Using Boosting**

The ranking for the features (“weak learners”) found by our boosting version is as follows: length of eye brow (average of left and right), length from neck to chin (average of left and right side), eigen coefficient-2, eigen coefficient-4, length of nose (average of left and right half), length of ear lobe (average of left and right side), frontal half of neck circumference, eigen coefficient-5, circumference of mouth, length of face (lower part), eigen coefficient-7, inter eye distance, distance from chin to mouth (mid point), inter ear distance, eigen coefficient-10, outer circumference of eye (average of left and right eyes), eigen coefficient-11, distance from mouth to nose bottom tip, length from shoulder to neck (average of left and right side), eigen coefficient-9, circumference of nose (lower part), eigen coefficient-13, eigen coefficient-17, eigen coefficient-12, eigen coefficient-18, distance from lower forehead to hair line. Ranking using boosting (see above) is different from ranking of anthropometric features using similarity (see Table 3).
using anthropometric and appearance-based features reconfirms the importance of the eyebrow for human authentication [88, 89] and ranks it as the top feature. The use of measurements outside the face, e.g., neck and shoulders helps too. Note that the ranks for two “apparently” similar features, the length from neck to chin (average of left and right side) (“curve”) and distance from chin to mouth (mid point) (“straight line”) (see Figure 18) are found by boosting to be quite different, e.g., ranks #2 and #12, respectively.

Decision-Level Fusion Using Backpropagation

The standard Euclidean distance is used as the similarity measure $S$ for both appearance-based and anthropometric-based methods to find the closest match in the gallery for a given probe image. One obtains similarity scores for using PCA, Fisherfaces (PCA + LDA), and anthropometric methods independent of each other. A weighted fusion scheme is used with the combination weights obtained by training a backpropagation artificial neural network (ANN). Assuming that $S_{PCA}$, $S_{(PCA+LDA)}$, and $S_{GEO}$ are the similarity score using PCA, PCA + LDA, and the (anthropometric-based) geometric methods, respectively, and $W_{PCA}$, $W_{(PCA+LDA)}$, $W_{GEO}$ are the corresponding weights, weighted hybrid similarity distances are computed as:

**Equation 33**

$$S_{HYBRID} = W_{PCA} \times S_{PCA} + W_{GEO} \times S_{GEO}$$

**Equation 34**

$$S_{HYBRID} = W_{(PCA+LDA)} \times S_{(PCA+LDA)} + W_{GEO} \times S_{GEO}$$
A distinct data set (from those used for enrollment and testing) is used to derive the weights. ANN training takes place using MATLAB’s Neural Network toolbox. The weights obtained are $W_{\text{PCA}} = 0.365$ ($W_{\text{GEO}} = 0.635$) and $W_{(\text{PCA+LDA})} = 0.312$ ($W_{\text{GEO}} = 0.688$). We report next the results from two sets of experiments conducted using FERET, AR and Essex face databases.

4.4.3 Experiments

The experimental results presented in this section show the feasibility and robustness of the hybrid methodology. The robustness is vis-à-vis disguise, occlusion, varying illumination, and temporal changes including varying facial expressions. Data sets and results are reported next.

Data Sets

Two sets of experiments were conducted using FERET, AR and Essex databases. The first series of experiments, Part I, is mostly about feasibility, with large scale and diversity aspects deferred to a second series of experiments – Part II. Details regarding datasets employed can be seen in section 4.3.3.

Results – Part I

Part-I experiments employ 3 types of face images from 30 subjects: clean, disguised, and occluded. From the two databases, 90 clean face images are chosen, 30 of which display a neutral expression, 30 show varying illumination, and 30 show varying facial expression. 90 additional disguised images using artificially created beards are generated using Adobe’s Photoshop. 90 additional occluded images are created by masking half of the face in a clean face image again using Adobe’s Photoshop. The
available corpora thus include 270 images (90 clean, 90 disguised, and 90 occluded) from 30 subjects. All the images used belong to male subjects. These images are used to create 3 distinct sets: ANN training (see section 4.4.2), gallery, and probe sets, respectively. 200 additional images from the AR database are used to generate the appearance-based face space. The Face Identification and Evaluation Software [83], developed by Colorado State University (CSU) (http://www.cs.colostate.edu/evalfacerec/index.html), is used to assay the appearance-based method, PCA and PCA + LDA. To meet CSU software requirements the raw images from the AR database and JPEG images from the Essex database are converted to PGM format using the ImageMagick library (http://www.imagemagick.org/script/index.php). The eye coordinates are manually obtained using MATLAB getpts function. All the images used for PCA and PCA + LDA experiments are resized to 96 x 72 pixels in size. Finally, a five-step normalization using CSU software is carried out for (i) converting 256 gray levels into floating points; (ii) geometric normalization that lines up the chosen eye coordinates; (iii) cropping the image using an elliptical mask; (iv) equalizing the histogram of the unmasked part of the image; and (v) normalizing the pixel values to mean zero and variance of one. PCA basis eigenvectors are computed using 200 images from the AR. The final base retained (for building biometric signatures) is spanned using a reduced 120 dimensional PCA subspace.

The Euclidean distance and nearest neighbor classifier are used for matching and human authentication. The experiments conducted are of the following type: Simple-1) appearance-based recognition using PCA + LDA utilizing top 60% eigenfaces; Simple-2)
appearance-based recognition using PCA + LDA with 90% energy PCA cutoff; Simple-3) appearance-based recognition using PCA with top 60% eigenfaces, Simple-4) appearance-based recognition using PCA with 90% energy cutoff; Hybrid-1) hybrid ANN recognition using PCA + LDA and anthropometric features from face only; Hybrid-2) hybrid ANN recognition using PCA + LDA and holistic anthropometric features from face, head and shoulders; Hybrid-3) hybrid ANN recognition using PCA and anthropometric features from face only; Hybrid-4) hybrid ANN recognition using PCA and holistic anthropometric features from face, head and shoulders; and, Hybrid-5) hybrid recognition that combines eigen (PCA) coefficients and anthropometric features using boosting. The hybrid methods 1 and 3 use 9 anthropometric features from face only while the hybrid methods 2 and 4 use 13 anthropometric features (see Table 3). The performance for the hybrid methods 1, 2, 3, and, 4 is obtained using the combination weights derived earlier (see section 4.4.2). Feature level fusion using all the 19 anthropometric features and eigen coefficients (subject to 90% energy PCA cutoff) are fed to the hybrid “boosting” method 5. The top ranked 25 features are used to build the strong boosting classifier. Experiments are conducted for 2 data combinations, gallery (clean) / probe (occluded) and gallery (clean) / probe (disguised). Experiments are also conducted on PCA and PCA + LDA using clean images for both gallery and probes to establish “clean” benchmarks against disguised / occluded biometric face conditions.

PCA consistently performs better than Fisherfaces, both on its own and as part of hybrid methods. Figures 19 and 20, and Tables 4 and 5, therefore omit the results obtained using Fisherfaces. Experimental results tabulate the recognition rates for
disguised and occluded biometric face conditions in Tables 4 and Table 5, respectively. Receiver Operating Characteristic (ROC) curves are generated to show the performance for some of the methods used, namely, PCA with top 60% eigenfaces (Simple-3), PCA with energy cutoff (Simple-4), PCA + Anthropometric Face (Hybrid-3), PCA + Anthropometric Holistic (Hybrid-4) and PCA + Anthropometric using boosting (Hybrid-5). The ROC curves are shown in Figure 19 for disguised (with beard) images and in Figure 20 for occluded (half-face) images. The performance of PCA on clean face images is also shown in Figures 19 and 20 for comparison purposes. The results show that disguise and occlusion lead to a significant deterioration in performance for both PCA and PCA + LDA. ROC results also show that the proposed hybrid methods significantly improve the performance of PCA and PCA + LDA methods both on disguised and occluded images. Note that authentication using appearance-based PCA outperforms PCA + LDA. Among the appearance-based recognition approaches, those utilizing the 90% energy PCA cutoff criteria yield better performance compared to those using the top 60% eigen vectors suggested by CSU. Hybrid methods using holistic anthropometric features outperform hybrid methods that employ face measurements only. One is thus led to conclude that the use of features below the face enhances human authentication. The hybrid-5 method, which employs the top 25 features found by boosting, is the top human authentication performer with accuracies of 76% and 56% for disguised and occluded images, respectively, at FAR = 10%. The experiments conducted showed a higher improvement rate for occluded images compared to disguised images (because occlusion is localized compared to disguise). The performance reported surpasses that obtained
using the worst performer Fisherfaces (see Table 6) for disguised and occluded images, and is similar to recent (and realistic) results obtained on Facebook (using clean data) (see section 4.2). Another observation made is that performance for human authentication rate saturates at FAR = 0.1.

Figure 19 Hybrid - ROC for Disguised Images
Figure 20 Hybrid - ROC for Occluded Images

Table 4 Hybrid - Recognition (Hit) Rates for Disguised (Face with Beard) Images

<table>
<thead>
<tr>
<th>FAR</th>
<th>Recognition (hit) rates</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Simple-1 [PCA + LDA] top 60%</td>
</tr>
<tr>
<td>0.05</td>
<td>0.2</td>
</tr>
<tr>
<td>0.1</td>
<td>0.33</td>
</tr>
<tr>
<td>0.15</td>
<td>0.64</td>
</tr>
<tr>
<td>0.2</td>
<td>0.64</td>
</tr>
</tbody>
</table>

Table 5 Hybrid - Recognition (Hit) Rates for Occluded (Half-Face) Images

<table>
<thead>
<tr>
<th>FAR</th>
<th>Recognition (hit) rates</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Simple-1 [PCA + LDA] top 60%</td>
</tr>
<tr>
<td>0.05</td>
<td>0.1</td>
</tr>
<tr>
<td>0.1</td>
<td>0.21</td>
</tr>
<tr>
<td>0.15</td>
<td>0.31</td>
</tr>
<tr>
<td>0.2</td>
<td>0.42</td>
</tr>
</tbody>
</table>
Results – Part II

The second series of experiments (Part II) aims to show that feasibility, robustness, and utility for our proposed methods cover additional degrees of freedom: (a) face space derived using FERET while training (enrollment) and testing take place on another database AR using naturally (scarves and glasses) occluded images (see Figure 21); (b) large scale aspects and diversity, with AR gallery now consisting of 100 subjects, while gender diversity covers 60 males and 40 females; and (c) expanded functionality to allow enrollment of occluded faces with testing taking place on clean faces. The experiments described here have access to two data sources, the gray scale FERET database (http://www.itl.nist.gov/iad/humanid/feret/feret_master.html) and the AR database. The FERET database [80] contains about 15,000 images with varying facial expression, illuminations and side poses, while the AR database contains over 4,000 images. The datasets used for conducting experiments are summarized in Table 7.

### Table 6 Hybrid - Improvements in Recognition Rates for Disguised and Occluded Images

<table>
<thead>
<tr>
<th>FAR</th>
<th>% Improvement in recognition rate compared to PCA + LDA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disguised images</td>
</tr>
<tr>
<td></td>
<td>Hybrid-1</td>
</tr>
<tr>
<td>0.05</td>
<td>100.0</td>
</tr>
<tr>
<td>0.1</td>
<td>22.6</td>
</tr>
<tr>
<td>0.15</td>
<td>19.9</td>
</tr>
<tr>
<td>0.2</td>
<td>12.5</td>
</tr>
</tbody>
</table>
As before the 5-step normalization for face images is performed using the CSU software. Experiments were conducted using top 60% eigenfaces, which resulted in 1200-D face space, and using 90% energy criteria, which resulted in 45 eigenfaces. The anthropometric feature selection using these data sets resulted in the same ranking as tabulated in Table 3. The weights for the decision level fusion obtained from ANN were $W_{PCA} = 0.42$ ($W_{GEO} = 0.58$) and $W_{(PCA+LDA)} = 0.39$ ($W_{GEO}=0.61$). For the boosting method, feature level fusion using all the 19 anthropometric features and 45 eigenfaces coefficients (subject to 90% energy PCA cutoff) were utilized. The top ranked 25 features were used to build the strong boosting classifier. Experiments were conducted both for the gallery (clean) / probe (occluded) and gallery (occluded) / probe (clean) data combinations.
Results for the series of experiments - Part II are tabulated in these Tables (Table 8, Table 9, Table 10 and Table 11) and the ROC curves are plotted in these Figures (Figure 22, Figure 23, Figure 24 and Figure 25). As one can see, the results show performance similar or better with that obtained in the earlier series of experiment - Part I. Holistic anthropometric yields better performance than face only anthropometric measures. The hybrid-5 boosting method yields the best performance overall. Recognition rate of 80% is obtained for the train (clean) / test (occluded with sunglasses) combination at FAR = 10%. The new ranking of the features found by the boosting method is as follows: (again top ranked) length of eye brow (average of left and right), length from neck to chin (average of left and right side), eigen coefficient-1, eigen coefficient-3, eigen coefficient-9, length of nose (average of left and right half), length of ear lobe (average of left and right side), eigen coefficient 6, frontal half of neck circumference, circumference of mouth, eigen coefficient 17, length of face (lower part), eigen coefficient 19, inter eye distance, distance from chin to mouth (mid point), eigen coefficient 23, inter ear distance, outer circumference of eye (average of left and right eyes), eigen coefficient 31, distance from mouth to nose bottom tip, length from shoulder to neck (average of left and right side), circumference of nose (lower part), eigen coefficient 14, eigen coefficient 21, distance from lower forehead to hair line. The observation made for series of experiments – Part I, that performance for face recognition rate saturates at FAR = 0.1, holds true here too.
### Table 8 Hybrid - Recognition (Hit) Rates: Train (Clean)/Test (Occluded-Sunglasses)

<table>
<thead>
<tr>
<th>FAR</th>
<th>Recognition (hit) rates</th>
<th>Simple-1 PCA + LDA (top 60%)</th>
<th>Simple-2 PCA + LDA, E (90% energy)</th>
<th>Simple-3 PCA (top 60%)</th>
<th>Simple-4 PCA, E (90% energy)</th>
<th>Hybrid-1 PCA + LDA, E + Anth., Face</th>
<th>Hybrid-2 PCA + LDA, E + Anth., Holistic</th>
<th>Hybrid-3 PCA, E + Anth., Face</th>
<th>Hybrid-4 PCA, E + Anth., Holistic</th>
<th>Hybrid-5 Boosting</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>0.41</td>
<td>0.41</td>
<td>0.42</td>
<td>0.42</td>
<td>0.47</td>
<td>0.49</td>
<td>0.53</td>
<td>0.56</td>
<td>0.59</td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td>0.58</td>
<td>0.59</td>
<td>0.60</td>
<td>0.61</td>
<td>0.70</td>
<td>0.73</td>
<td>0.76</td>
<td>0.77</td>
<td>0.80</td>
<td></td>
</tr>
<tr>
<td>0.15</td>
<td>0.65</td>
<td>0.65</td>
<td>0.68</td>
<td>0.68</td>
<td>0.72</td>
<td>0.76</td>
<td>0.78</td>
<td>0.79</td>
<td>0.81</td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td>0.65</td>
<td>0.65</td>
<td>0.69</td>
<td>0.69</td>
<td>0.72</td>
<td>0.76</td>
<td>0.78</td>
<td>0.79</td>
<td>0.81</td>
<td></td>
</tr>
</tbody>
</table>

### Table 9 Hybrid - Recognition (Hit) Rates: Train (Clean)/Test (Occluded-Scarf)

<table>
<thead>
<tr>
<th>FAR</th>
<th>Recognition (hit) rates</th>
<th>Simple-1 PCA + LDA (top 60%)</th>
<th>Simple-2 PCA + LDA, E (90% energy)</th>
<th>Simple-3 PCA (top 60%)</th>
<th>Simple-4 PCA, E (90% energy)</th>
<th>Hybrid-1 PCA + LDA, E + Anth., Face</th>
<th>Hybrid-2 PCA + LDA, E + Anth., Holistic</th>
<th>Hybrid-3 PCA, E + Anth., Face</th>
<th>Hybrid-4 PCA, E + Anth., Holistic</th>
<th>Hybrid-5 Boosting</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>0.25</td>
<td>0.28</td>
<td>0.27</td>
<td>0.33</td>
<td>0.41</td>
<td>0.41</td>
<td>0.42</td>
<td>0.42</td>
<td>0.50</td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td>0.31</td>
<td>0.31</td>
<td>0.32</td>
<td>0.37</td>
<td>0.46</td>
<td>0.46</td>
<td>0.48</td>
<td>0.48</td>
<td>0.59</td>
<td></td>
</tr>
<tr>
<td>0.15</td>
<td>0.36</td>
<td>0.38</td>
<td>0.385</td>
<td>0.39</td>
<td>0.52</td>
<td>0.53</td>
<td>0.56</td>
<td>0.57</td>
<td>0.62</td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td>0.43</td>
<td>0.43</td>
<td>0.44</td>
<td>0.45</td>
<td>0.59</td>
<td>0.59</td>
<td>0.60</td>
<td>0.61</td>
<td>0.63</td>
<td></td>
</tr>
</tbody>
</table>

### Table 10 Hybrid - Recognition (Hit) Rates: Train (Occluded-Sunglasses)/Test (Clean)

<table>
<thead>
<tr>
<th>FAR</th>
<th>Recognition (hit) rates</th>
<th>Simple-1 PCA + LDA (top 60%)</th>
<th>Simple-2 PCA + LDA, E (90% energy)</th>
<th>Simple-3 PCA (top 60%)</th>
<th>Simple-4 PCA, E (90% energy)</th>
<th>Hybrid-1 PCA + LDA, E + Anth., Face</th>
<th>Hybrid-2 PCA + LDA, E + Anth., Holistic</th>
<th>Hybrid-3 PCA, E + Anth., Face</th>
<th>Hybrid-4 PCA, E + Anth., Holistic</th>
<th>Hybrid-5 Boosting</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>0.39</td>
<td>0.40</td>
<td>0.40</td>
<td>0.41</td>
<td>0.44</td>
<td>0.48</td>
<td>0.50</td>
<td>0.53</td>
<td>0.54</td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td>0.56</td>
<td>0.58</td>
<td>0.59</td>
<td>0.60</td>
<td>0.65</td>
<td>0.68</td>
<td>0.69</td>
<td>0.75</td>
<td>0.76</td>
<td></td>
</tr>
<tr>
<td>0.15</td>
<td>0.62</td>
<td>0.62</td>
<td>0.63</td>
<td>0.64</td>
<td>0.70</td>
<td>0.71</td>
<td>0.72</td>
<td>0.75</td>
<td>0.76</td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td>0.62</td>
<td>0.63</td>
<td>0.63</td>
<td>0.64</td>
<td>0.70</td>
<td>0.71</td>
<td>0.72</td>
<td>0.75</td>
<td>0.76</td>
<td></td>
</tr>
</tbody>
</table>

### Table 11 Hybrid - Recognition (Hit) Rates: Train (Occluded-Scarf)/Test (Clean)

<table>
<thead>
<tr>
<th>FAR</th>
<th>Recognition (hit) rates</th>
<th>Simple-1 PCA + LDA (top 60%)</th>
<th>Simple-2 PCA + LDA, E (90% energy)</th>
<th>Simple-3 PCA (top 60%)</th>
<th>Simple-4 PCA, E (90% energy)</th>
<th>Hybrid-1 PCA + LDA, E + Anth., Face</th>
<th>Hybrid-2 PCA + LDA, E + Anth., Holistic</th>
<th>Hybrid-3 PCA, E + Anth., Face</th>
<th>Hybrid-4 PCA, E + Anth., Holistic</th>
<th>Hybrid-5 Boosting</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>0.21</td>
<td>0.22</td>
<td>0.24</td>
<td>0.28</td>
<td>0.29</td>
<td>0.35</td>
<td>0.35</td>
<td>0.36</td>
<td>0.36</td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td>0.28</td>
<td>0.29</td>
<td>0.30</td>
<td>0.32</td>
<td>0.34</td>
<td>0.38</td>
<td>0.39</td>
<td>0.40</td>
<td>0.40</td>
<td></td>
</tr>
<tr>
<td>0.15</td>
<td>0.30</td>
<td>0.30</td>
<td>0.32</td>
<td>0.33</td>
<td>0.35</td>
<td>0.39</td>
<td>0.41</td>
<td>0.41</td>
<td>0.41</td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td>0.30</td>
<td>0.30</td>
<td>0.32</td>
<td>0.34</td>
<td>0.35</td>
<td>0.39</td>
<td>0.42</td>
<td>0.43</td>
<td>0.42</td>
<td></td>
</tr>
</tbody>
</table>
Figure 22 Hybrid - ROC Curve for Occluded (Sunglasses) Images
Figure 23 Hybrid - ROC Curve for Occluded (Scarf) Images
Figure 24 Hybrid - ROC Curve for Clean Images (Train-Sunglasses)
4.5 Summary

Two methods for face recognition, namely, ARCF and a hybrid anthropometric and appearance-based approach were developed. ARCF utilized recognition-by-parts strategy. ARCF provide information that involves both appearance and location. The cluster and strength of the ARCF correlation peaks indicate the confidence of the face authentication made, if any. The development of ARCF, motivated by MACE filters and adaptive beam-forming from radar / sonar, is driven by Tikhonov regularization. The adaptive aspect of ARCF comes from their derivation using both training and test data, similar to transduction, while the robust aspect benefits from the correlation peak.
optimization to decrease their sensitivity to noise and distortions. The comparative advantages of ARCF are motivated, explained, and illustrated vis-à-vis existing correlation filters. Experimental evidence shows the feasibility and reliability of ARCF vis-à-vis occlusion, disguise, and illumination, expression, and temporal variability. The generalization ability of ARCF is also illustrated when decision-making thresholds learned a priori from one data base, e.g., FERET, carry over to face images from another database, e.g., AR.

The hybrid methodology developed here employs feature-level fusion using boosting and decision-level fusion using neural networks for the purpose of robust human authentication vis-à-vis face occlusion and disguise. Holistic anthropometric and appearance-based features feed the data fusion stage. In addition to standard head and face geometric measurements, the proposed holistic anthropometric features extract additional measurements below the face, which describe the neck and shoulder and their contextual relations to head and face. The appearance-based features include standard PCA or Fisherfaces. Experimental data shows the feasibility and utility of the proposed hybrid (extended geometry + appearance) approach for robust human authentication vis-à-vis occluded and/or degraded face biometrics. The authentication results presented compare favorably against both appearance-based methods and hybrid methods with anthropometric features confined to face and head.
5 PHISHING DETECTION

The novel phishing detection methodologies developed in this research are presented in this chapter. The motivation for this research is presented in section 5.1. A review of state-of-the art research is presented in section 5.2. The phishing email detection methodology that employs PLSA, AdaBoost and Co-Training is presented in section 5.3. The phishing website detection methodology that employs methods LDA and AdaBoost is presented in section 5.4. Section 5.5 presents the impersonated entity discovery methodology that employs CRF. The methodologies developed are summarized in section 5.6.

5.1 Motivation

Stealing a person’s identity is one of the most profitable crimes committed by criminals. Among 1.3 million complaints received by the Federal Trade Commission in 2009, identity theft ranked first and accounted for 21% of the complaints costing consumers over 1.7 billion US dollars [1]. Identity theft has been around for many years while the means of committing it has changed with technology. The traditional way criminals steal a person’s identity is by killing the individual. Another way to steal identity is using phone scams, where, criminals inform the person that they have won a sweepstake, and convince the user to reveal some personal information to claim the money. The more popular method of identity theft that is prevalent even today is called
Dumpster Diving. When people discard letters, financial records, and other personal information in the garbage dump without shredding, criminals scavenge those dumps looking for sensitive information such as credit card, bank account social security numbers, and use that information to commit crimes.

With the advent of Internet, the most popular way to steal identity is through “phishing”. Like in traditional fishing where fishermen troll the river in a boat to catch fish, in “phishing”, attackers troll the Internet using email message with convincing content as bait to steal users personal information. The email directs the user via a hyperlink to a website owned by criminals that looks very similar to a legitimate website. The user will then be asked to enter personal and financial information either to update existing information or to purchase a product. In reality, this lets the criminal to have access to that valuable information which they then use to commit fraud or to sell it to a bidder. Phishers can also trick users into downloading malicious codes or malware after they click on a link embedded in the email. This is a useful tool in crimes like economic espionage where sensitive internal communications can be accessed and trade secrets stolen. Phishing has been around since 1996 but has become more common and more sophisticated. Recent phishing attack on the Gmail system stole emails of US government officials, contractors, and military personnel [90].

Considerable research has been done towards protecting users from phishing attacks. They include firewalls, black listing certain domains and Internet protocol (IP) addresses, spam filtering techniques, client side toolbars, and user education. Each of these existing techniques has some advantages and some disadvantages. For example,
existing filters have misclassification rates, the blacklist approach is harder to maintain with every expanding IP address/domain space, while the user ignores client side toolbar warnings.

The main contributions of this research are as follows: (i) Phishing email detection methodology, which employs Probabilistic Latent Semantic Analysis (PLSA), AdaBoost and Co-Training. The methodology handles synonyms (multiple words with similar meanings), polysemy (words with multiple meanings), intentionally misspelled words and other linguistic variations found in phishing. In addition, the methodology requires only a small percentage of data be annotated thus saving time, labor, and avoiding errors incurred in human annotation; (ii) Phishing website detection methodology, which employs Latent Dirichlet Allocation (LDA) and AdaBoost. The content driven methodology is robust to changes in word usage. It is device neutral (can be applied to desktop and mobile devices) and language neutral (can be applied to content in different languages); and, (iii) Impersonated entity discovery methodology, which employs LDA, AdaBoost and Condition Random Field (CRF). The methodology automatically extracts the entity the attacker is trying to portray. This helps service providers to collaborate with each other to exchange attack information and protect their customers. The legitimate organization can take down the offending site thus preventing its customers from falling for phishing, which in turn leads to satisfied customers.

5.2 Background

Several methods have been developed to protect users from phishing attacks. In this section, protection strategy has been organized based on where in the attack flow that
technique belongs (see Figure 26). State-of-the-art tools, developed by earlier researchers, that employ the corresponding technique and their advantages and disadvantages are summarized in this section. In this research, the state-of-the-art review is limited to attacks that are initiated via email. However, most of these are applicable to other modes of attacks such as social network sites, blogs, etc.

**Figure 26 Phishing Protection Techniques**

**Network Level Protection**

The network level protection is typically achieved by blocking range of IP addresses or a set of domains from entering the network. DNSBL [91] is the database used by several internet service providers. This list is updated with new address after observing for a period of time abusive behavior. Hence, this approach is reactive. Attackers evade this protection technique by hijacking legitimate user’s PC and
constantly moving from one IP to another IP address. Snort [92] is a open source software that is employed at network level. Rules to enforce protection must constantly be manually updated. Kim and Huh [93] compared four different classification techniques to detect DNS-poisoning-based phishing attacks using routing information gathered over 1-week period. Authors observed that k-nearest neighbor algorithm achieved best results with a false positive rate of 0.7% and true positive rate of 99.4%.

**Authentication**

There are two levels of authentication, user level and domain level. Typically, user is authenticated by the email service provider, before he or she sends an email (user level). The domain level authentication is performed in the provider-provider communication (one mail server to the other mail server). The user level authentication is performed using password as credential. The password authentication can easily be cracked as evidenced by surge in phishing attacks. Microsoft has developed a technology called Sender ID [94] while Yahoo has a similar technology called DomainKey [95]. Both these technologies perform domain level authentication. In order for these domain level technologies to work, providers on the sender and the recipient side must implement. Due to lack of agreement between email providers, this technology is not that prevalent.

**Server Side Filters and Classifiers**

Server side filters and classifiers typically extract features from the email and train a classifier to classify phishing email versus non-phishing email. Classifiers can be trained directly on various features extracted from the data or by applying dimensionality
reduction techniques before training the classifier. Kim et al. [96] applied three
dimensionality reduction methods, namely, Centroid, Orthogonal Centroid, and Linear
Discriminant Analysis and tested their effectiveness on three different classifiers: Support
Vector Machines, k-Nearest Neighbor, and Centroid-based classification. Authors
concluded that dimension reduction techniques achieve high efficiency without
sacrificing prediction accuracy. SpamAssassin [97] is a widely used host-level filter. This
is a rule-based filter that requires constantly changing for the rule to be effective.
Attackers figure out the rule being employed and bypass these filters by appropriately
constructing the email. PILFER [98] is another email classifier that is trained using ten
features extracted from email data. Both these filters have high misclassification rates.
Pamunuwa et al. [99] developed two-stage host based intrusion detection system for
detecting phishing detection that included a recursive crawler for capturing websites from
email message links. Abu-Nimeh et al. [100] presented a comparative evaluation of
classification techniques such as Logistic Regression, Bayesian Adaptive Regression
Trees, Support Vector Machines, Random Forests, and Neural Network. Authors trained
the classifier using 43 features on a private ham email and public phishing email data and
showed that random forest outperformed other classifiers when weighted equally but
resulted in worst false positive rate. Neural network had the highest Area Under ROC
Curve. Later work by Abu-Nimeh et al. [101] developed a method to detect phishing
using Bayesian Additive Regression Trees and obtained better prediction than their
erlier work. Miyamoto et al. [102] did a similar comparison of machine learning
techniques for phishing website detection using about 3,000 website data. They obtained
F-measure of 0.85 using AdaBoost. Toolan and Carthy [103] classified emails using C5.0 algorithm and ensemble of different classifiers. Authors obtained an F-measure of 99.31% using the publicly available dataset (PhishingCorpus and SpamAssassin) of 8 K emails. Gansterer and Pölz [104] developed a feature-based classifier or ternary classification, spam versus phish versus good. Authors utilized 11,000 phishing emails from a proprietary data source and publicly available TREC corpus for good and spam and obtained a classification accuracy of 97%. Bergholz et al. [105] trained a classifier using features obtained using Dynamic Markov Chain and Class-Topic Models. Authors obtained results better than PILFER on the same public corpus and showed effectiveness of topic features. Later work by Bergholz et al. [106, 107] included additional features such as identification of hidden salting, embedded logos and external links and evaluated on a proprietary real life data from a commercial internet provider of size 40 K. Toolan and Carthy [108] proposed and ranked 40 different features using the information gain criteria. Khonji et al. [109] did an evaluation of feature selection algorithms and feature subset search methods on the same public corpus that most of the other research has been conducted. The study showed feature subset of 21 heuristic features yielded F-measure of 99.39%. Al-Momani et al. [110] achieved classification accuracy of 99.7% by applying a clustering algorithm for phishing detection while Zhan and Thomas [111] obtained a true positive rate of 99% by applying Stochastic Learning Weak Estimation approach. Yearwood et al. [112] obtained profiles of phishing activity by solving the problem using a multi-class classification problem utilizing features extracted from URLs in the emails. This study is closely related to Bergholz et al. [105, 106], in the sense that, we use topic
models PLSA and LDA (as compared to CLTOM) for phishing detection. However, our topic model is built to account for intentional misspelling and uses part-of-speech (verbs, nouns, adjectives, and adverbs) to build the model. We also show the effectiveness of our method on a large corpus of unlabeled data using Co-Training.

Several research has been done for phishing website detection. Xiang et al. [113] proposed a layered anti-phishing approach for detecting phishing web sites. Authors used a comprehensive feature-based detection algorithm to detect and filter out non-login form web pages and achieved 92% true positive rate and 0.4% false positive rate. Khonji et al. [114] proposed a technique for detecting phishing website by lexically analyzing URL tokens. Authors evaluated 70 K phishing URLs and obtained classification accuracy of 97%. Zhang et al. [115] proposed a text classifier, image classifier, and an algorithm that fused the two-classifier results to detect phishing web page detection and they concluded that fusion outperformed the performance of individual classifiers. Hsu et al. [116] proposed a solution for phish URL detection using suffix tree clustering methodology while Khonji et al. [117] proposed a heuristic solution. Wenyin et al. [118] developed an approach to detect phishing target from the content of the webpage. Most of the above research is limited to website detection; however, we propose a generic content-based approach that can be applied to email, web pages, blogs, and social networking posts.

Client Side Tools

Tools that operate on the client side (i.e., user’s machine) include user profile filters and browser-based toolbars. SpoofGuard [119], CatchingPhish [120], CallingID [121], CloudMark [122], NetCraft [123], FirePhish [124], eBay toolbar [125], and IE
Phishing Filter [126] are some of the client side tools. User profile filters observe user’s website visiting pattern and maintain a list of URLs in local database. When a user visits a URL that is different from his/her website visits, it warns the user with a dialog. Toolbars are built and trained using the typical pattern of phishing website URLs. Some patterns of phish website URLs include IP address in the URL, long URLs, many dots in the URL, etc. This technique is very susceptible to technology changes (such as IPV4 versus IPV6, tiny URLs) and hence it is not robust. Moreover, most users do not pay attention to the warning dialogs and hence it is not an effective protection technique. Abu-Nimeh and Nair [127] presented a new attack using DNS poisoning that bypass the client side toolbars. Their evaluation of seven tools concluded that none of them were able to detect the attack there by making these tools ineffective. Jain and Richariya [128] implemented a prototype web browser to detect phishing URLs. Authors did not compare their implementation with other browser-based tools and hence the effectiveness of the tool is not clear. Lin et al. [129] evaluated domain highlighting, the approach where browser highlights the domain name in the address bar, and concluded that this approach cannot be relied upon solely to detect and prevent phishing attacks. Chen et al. [130] presented a scientific assessment of user interface design elements such as font type, color, message placement, icon type, etc., used in various tools and concluded that existing tools fail to consider preference of the user while displaying warning and errors. Author’s findings conclude that users prefer customization and personalization of these tools. Felt and Wagner [131] examined the threat of phishing on mobile devices. Authors analyzed 100 mobile applications and 85 web sites and found that attackers can spoof
mobile web site. Authors found that Android and Apple-sponsored sites are top phish targets.

**Prevent Against Duplication**

This technique involves making the legitimate website harder to reproduce. In all the legitimate website, the login page is not protected. Hence, an attacker can easily copy the code, styles, graphics and HTML to create a fake website. Hence, a protection approach could be to make this copy harder. There is no earlier work done this area and hence it is ideal for future research.

**User Education**

Educating users about security is challenging, particularly in the context of phishing, as i) users are not motivated to read about security in general and therefore do not take time to educate themselves about phishing; ii) for most users, security is a secondary task; and iii) it is difficult to teach people to make the right online trust decision without increasing their tendency to misjudge non-threats as threats. The basic mode of educating user is posting help pages and websites warning the user about phishing. MailFrontier [132] has setup a website containing screenshots of several phishing emails. Robila and Ragucci [133] evaluated the effect of user education in differentiating phishing and good emails. The authors presented a lecture on how to identify phishing emails and the harm of falling for phishing in a class on introduction to computing. At the end of the lecture, the authors presented student with both phishing and good emails. Students were then asked to identify the email type. The study concluded that students identified phishing emails correctly after the lecture. Students
also acknowledged the usefulness of the lecture. Similar study was also conducted at the Indiana University [134]. Arachchilage and Cole [135] designed an educational mobile game for home computers to protect users from phishing attacks. The game was designed to educate users to recognize phishing URLs. Authors developed a prototype simulator using Google App Inventor Emulator. Tseng et al. [136] also designed a game to educate users about phishing based on the content of the website. Moore and Clayton [137] conducted a study of how attackers discover potential hosts for phishing websites and concluded that search engine as one primary source. Authors of the study concluded public disclosure of phishing sites, such as the one done by phishtank.com, significantly reduces host compromise by attackers. Users are the weakest link in the cybersecurity. Most users do not have a secure password. A large percentage of users use their login names as passwords or choose passwords that are prone to dictionary attacks or write down their passwords. They also use outdated security software, click on links that install malware and spyware and visit and disclose their credentials to phishing sites. Hence, user education is one of the key components of cybersecurity. Microsoft invests millions of dollars towards user education by conducting workshops, webinars, awareness campaigns and hosting help pages. Kumaraguru et al. [138] developed an email-based anti-phishing education system called ‘PhishGuru’ and an online game called ‘Anti-Phishing Phil’ that teaches users how to use cues in URLs to avoid falling for phishing attacks. In PhishGuru, users are periodically sent training emails in the form of simulated phishing emails, either by the system administrators or from a training company. Users access these training emails in their inbox while they are checking their regular emails.
These training emails look just like phishing emails, urging people to go to some website and login. If people fall for the training email (that is, they click on a link in that email), PhishGuru provides an intervention message that explains that they are at risk for phishing attacks and give some tips to users for protecting themselves. Authors concluded that providing immediate feedback at this teachable moment enhances learning. In Anti-Phishing Phil, users are motivated to learn by embedding training into a fun activity. The highly interactive nature of the game enables users to learn to distinguish legitimate links from fraudulent ones. Anti-Phishing Phil complements PhishGuru by providing an entertaining platform for the rapid repetition and feedback needed to teach more difficult anti-phishing procedures. Authors of these systems concluded that automated phishing detection mechanisms should be the first line of defense against phishing attacks, and, user education offers a complementary approach.

Though several methods have been developed to detect and protect users from phishing attacks, these methods are not robust. Network level protection using domain and IP address blacklisting require periodic updates and are reactive in nature as list can be updated only after observing abuse pattern for some time period. Moreover, attackers compromise legitimate user’s machine to conduct phishing attacks and hence blacklisting may block legitimate user from using the web. Existing server side filters and classifiers result in high false positives and use feature sets that are susceptible to technology changes. The classifiers that use content for attack detection do not consider multiple meanings of a given word (synonyms) and different meanings of a word under different context (polysemy). Attackers make subtle changes to the text of the email by using
different words at different times and by using misspelled words to avoid detection by filters that does an exact word match. Thus, these filters often miss detection of phishing emails. Moreover, most of the existing mechanisms are not automated. Client side tools and filters expose the user one step closer to the attack. As users do not pay attention to warning dialogs, they end up falling for phishing attacks. Existing protection methods are ineffective in stopping the phishing attacks from reaching the end user. These methods are also susceptible to changes in technology. The goal of this research is to ‘stop the attack before it gets to the user’. The research plans to accomplish that by employing machine learning and natural language processing.

5.3 Phishing Email Detection

We propose here robust server side methodology to detect phishing emails, called phishGILLNET, which incorporates the power of natural language processing and machine learning. phishGILLNET is a multi-layered approach to detect phishing attacks. The first layer (phishGILLNET1) employs Probabilistic Latent Semantic Analysis (PLSA) (see section 3.3) to build a topic model. The topic model handles synonym (multiple words with similar meaning), polysemy (words with multiple meanings) and other linguistic variations found in phishing. Intentional misspelled words found in phishing are handled using Levenshtein editing and Google APIs for correction. Based on term document frequency (TDF) matrix as input PLSA finds phishing and non-phishing topics using tempered expectation maximization (TEM). The performance of phishGILLNET1 is evaluated using PLSA fold in technique and the classification is achieved using Fisher similarity. The second layer of phishGILLNET (phishGILLNET2)
employs AdaBoost (see section 3.2) to build a robust classifier. Using probability
distributions of the best PLSA topics as features the classifier is built using AdaBoost.
The third layer (phishGILLNET3) further expands phishGILLNET2 by building a
classifier from labeled and unlabeled examples by employing Co-Training (see section
3.2).

This section is organized as follows. The multi-layered phishing email detection
methodology phishGILLNET is presented in section 5.3.1. The architectural components
of phishGILLNET are presented in section 5.3.2. Experiments and results obtained on the
public corpus for each layer of phishGILLNET, namely, phishGILLNET1,
phishGILLNET2 and phishGILLNET3, are presented in section 5.3.3.

5.3.1 Methodology

A schematic representation of phishGILLNET is shown in Figure 27. Gillnetting
is a common fishing method used by fishermen in the ocean and in some freshwater areas
[139]. A ‘gillnet’, as the name implies, is a net that catches a fish by its ‘gill’. It is a layer
of netting hung vertically in the water by a float line on the top and a weighted line at the
bottom. The mesh size, depth and length of gillnet is determined by the species of fish
that fishermen is trying to catch. The net allows the head of the fish to pass through but
not its body. When the fish attempts to pass through, it gets stuck in the net by its gill and
could neither move forward nor backward. Just like a gillnet is used to catch a fish by its
gill, phishGILLNET is used to catch phishing attacks based on the linguistic variation in
the content.
phishGILLNET is a multi-layered methodology for detecting phishing attacks (Figure 28). Just like gillnet comes in various mesh sizes, the mesh size of the first layer of phishGILLNET (phishGILLNET1) is larger than the second layer (phishGILLNET2) and the second layer (phishGILLNET2) is larger than the third layer (phishGILLNET3). Phishing attacks missed by phishGILLNET1 are caught by phishGILLNET2 and the ones missed by phishGILLNET2 are caught by phishGILLNET3. All three layers of phishGILLNET employ PLSA (see section 3.3) to build a topic model that discovers phishing topics and non-phishing topics. phishGILLNET1 performs classification on unseen data using Fisher similarity function. phishGILNET2 builds a finer mesh utilizing PLSA topic features and AdaBoost (see Section 3.2). By employing PLSA, AdaBoost and Co-Training (see chapter 3), phishGILLNET3 further expands detection capability by building robust classifier from labeled and unlabeled data.

![Figure 27 phishGILLNET](image)
In order to build PLSA topic model, which all three layers of phishGILLNET employs, the methodology requires preparation of Term Document Frequency (TDF) matrix. Figure 29 shows the main components to build TDF, namely, Parser and TDF Matrix Builder. Both these components are described below:

**Parser**: Raw email data is typically present in Multipart Internet Mail Extension (MIME) format. phishGILLNET utilizes words and hyperlinks present in the body of the email to build PLSA model. Parser consists of the following:
Figure 29 phishGILLNET - Parsed and TDF Matrix Builder

**MIME Parser:** Parses email MIME message and extracts email headers and email body. Email body is further separated into HTML body part and text body part. For emails containing only text MIME part, the parser extracts text and hyperlinks. In a phishing email, these hyperlinks link to the phishing website.

**HTML Parser:** MIME message containing HTML body part is included as multipart/html part in the email body part. When the MIME parser detects a HTML part, it invokes the HTML parser to separate out text, style-sheets, hyperlinks and scripts. For the purpose of building PLSA model, both text and hyperlinks are considered.
Tokenizer: This tokenizes text present in email body and hyperlinks into separate words. Tokenizer utilizes white space (tabs, space, new lines) as token delimiters for the text. The hyperlinks are tokenized after replacing all non-alphanumeric characters with space.

**TDF Matrix Builder:** A term-document matrix describes the frequency of terms that occur in a collection of documents. The rows of the matrix correspond to document \((d_i)\) in the collection and the columns correspond to terms \((w_j)\) that present in those documents. For the text part, the terms \(w_j\) belong to one of the part-of-speech tags (adjectives, adverbs, nouns and verbs). For the hyperlinks part, all terms are used to build TDF. The matrix entries \(n(d_i, w_j)\) denotes number of times word \(w_j\) occurs in document \(d_i\).

Prior to building TDF Matrix the following pre-processing steps must be accomplished.

**Stop Words Removal:** Stops words are words that do not contain important significance for building the model. Some example stop words include the, at, like, etc. We remove stop words from all the tokenized email text.

**Stemming:** Stemming is a method for removing inflexional endings from certain words. For example, word ‘consigned’, after stemming becomes ‘consign’. Porter’s Stemming [140] algorithm is employed to stem words in email body.

**Dictionary Lookup:** WordNet [141] dictionary is employed to lookup words in dictionary. WorldNet database has Part of Speech (POS) extractor. It identifies verbs, nouns, adverbs and adjectives. Words found in WorldNet database forms part of the input for building TDF matrix using text. For the hyperlinks TDF, WordNet lookup and spell checker is skipped.
Spell Checker: Attackers intentionally misspell words in a phishing email to avoid detection by standard spam filters. For words that are not found in WordNet database, Google’s spell check API [142] is utilized to retrieve words that are similar to the misspelled word.

Levenshtein Distance: Levenshtein distance [143] is a metric for measuring the amount by which two words differs. The metric is also called edit distance. It is the minimum edit operations required to transform one word to another. The edit operations include insertion, deletion and substitution of a new character. In a phish email there are misspelled words, which after edit operation, is found in dictionary. Examples include “vuln’a’rability”, “youaccounts”, etc. Also, there are terms made of garbage characters that are never found in dictionary. We consider only misspelled words that can be corrected after certain edit operation. After obtaining the suggested words using Google API, Levenshtein distance is computed. Only those words whose edit distance is less than some configured threshold (default value of 5) are further included for building TDF matrix.

Build TDF Matrix: For email body text, using words, (specifically adjectives, adverbs, nouns and verbs), that found directly in dictionary and edited words using Levenshtein’ edit operation, the term-document-frequency matrix is created. For email hyperlinks, all terms are used to build TDF matrix.

Thus, phishGILLNET accounts for misspelled words, conjoined words and POS tags present in email body before building the TDF matrix. Once the TDF matrix is built using components described above, all three layers of phishGILLNET employs PLSA to
build the topic model for phishing detection. We present next the architecture of all three layers of phishGILLNET.

5.3.2 Architecture

The architecture of the multi-layered phishing email detection methodology phishGILLNET is presented in this section.

phishGILLNET1

phishGILLNET1 is the top layer of the multi-layered phishing detection methodology. It employs PLSA topic modeling technique to discover phishing and non-phishing topics and Fisher similarity function for classification. The architecture of phishGILLNET1 is shown in Figure 30. The architecture has four main components, parser, TDF matrix builder, PLSA model trainer, PLSA fold-in and Classifier. The architecture employs the parser to parse data and TDF matrix builder to build the TDF matrix (described in Section 5.3.1). It employs the PLSA modeling method to build the topic model. The implementation architecture of PLSA model trainer and PLSA fold-in are shown in Figure 31. The theory behind PLSA is described in Section 3.3.
Figure 30 phishGILLNET1 Architecture

Figure 31 phishGILLNET1 - PLSA Model Trainer and Fold-In
**PLSA Model Trainer:** The input to the model is the TDF matrix of the training data set. In this work, Tempered Expectation Maximization (TEM) algorithm described earlier in Section 3.3, was employed to build the topic model. PLSA algorithm is implemented using Java programming language.

**Initialization:** PLSA requires number of topics, $K$, to be specified at initialization similar to cluster analysis. The probability distributions are initialized using random numbers.

**E-Step:** The joint probability distribution values are computed using initialized probability distribution.

**M-Step:** In the M-step, word-topic and topic-document probabilities are computed using expressions given in PLSA model section.

**Compute Performance Metric:** Performance measure, log likelihood and perplexity are computed according to the equations given in the performance evaluation section (see section 3.4).

**PLSA Fold-In:** In fold-in, test data probability distributions are computed using the $P(w|z)$ value from the training phase as input. The TEM algorithm is employed to compute distributions on the test data set, while $P(w|z)$ is kept fixed.

**Classifier:** phishGILLNET1 categorizes email as phishing versus non-phishing using a similarity function. Using labeled emails as input data set, containing phishing emails and non-phishing emails, topic distribution probabilities and word distribution probabilities are obtained by building a PLSA model. The similarity score is computed using Fisher Kernel similarity function (see section 3.3) between test emails and emails in
the training set. The label of the training email that yields the highest similarity score is considered the label of the test email.

**phishGILLNET2**

phishGILLNET2 is a finer layer than phishGILLNET1. Instead of using Fisher similarity function for categorization using topic distribution probabilities, AdaBoost is employed to build a robust classifier using PLSA topic distribution probabilities as feature. Furthermore, phishGILLNET2 performs 3-class classification (phish, spam, good) as well as binary classification (phish, not phish).

The architecture of phishGILLNET2 is shown in Figure 32. phishGILLNET2 employs AdaBoost as the classifier ensemble. The PLSA topics are discovered as before in phishGILLNET1 and topic distribution probabilities on training data are estimated. AdaBoost classifier ensemble is built using these probabilities as features and several weak learners. Existing classification methods such as C4.5 decision tree, rule based classifier (RIPPER), random forest, support vector machines and logistic regression are used as the weak learners in phishGILLNET2. The performance of the classifier is compared using the metrics reported in Section 3.4. The open source software WEKA [86] was used for the implementation of phishGILLNET2.
phishGILLNET3 is the third layer of the multi-layered phishGILLNET. This layer employs AdaBoost and Co-Training algorithm to build a robust classifier using large corpus of unlabeled data. Labeling data to build classifiers require significant time and human labor. phishGILLNET3 eliminates the need for fully labeled corpus.

The architecture of phishGILLNET3 is shown in Figure 33. The motivation for this implementation is to evaluate the robustness of topic model, specifically PLSA, on a large corpus of unlabeled data. This architecture implements the Co-Training algorithm and applies to the email domain. The algorithm starts with small corpus of labeled emails (phishing and non phishing). Using parser components, email data is parsed into text.
present in the body of the email and hyper links. The text and hyper links form two views for applying the Co-Training algorithm. One of the assumptions behind the Co-Training algorithm is that the two views should not be perfectly co-related. In a phishing email, the text in body of the email will contain enticing content asking the user to click the hyperlink and the hyperlink and accompanying web content will contain the impersonating entity. There may be some correlation between the two views (body text and hyperlinks) but not perfect correlation. In addition, review of literature (see section 5.2) shows that classifiers built just using hyperlinks and just using body text yields good classification performance. Hence, we apply co-training to the email body text and hyperlink views. For the body text, all words in the email are used to build the PLSA model for text view. For the hyperlinks, terms are extracted by replacing all non-alphanumeric characters as token separator in the hyperlinks. These terms are used to build the PLSA model for the hyperlink view. For both views, once the PLSA model is built, the topic distribution probabilities are extracted as features. These features are used to build the classifier. The text classifier and the hyperlink classifier classify unlabeled email data and most confidently predicted email data is added to labeled corpus for the next iteration of co-training. The process repeats until there is no more unlabeled data to label.
5.3.3 Experiments

In this section, we present the details on experiments designed to build and evaluate phishGILLNET. This includes datasets employed, data preparation, training and test strategies and measures to evaluate performance.

Data Sets

Four publicly available email datasets and one publicly available phish URL data set were used to evaluate phishGILLNET. Email data sets include, (i) ham (good) emails from SpamAssassin corpus [144], (ii) phishing emails from the PhishingCorpus [145], (iii) good emails from Enron Email Dataset [146] and (iv) spam emails from SPAM Archive [147]. Phish URL data set includes (v) PhishTank [148].

SpamAssassin: SpamAssassin corpus contains a total of 6047 messages, of which, 4150 messages are good and the remaining are spam. These messages were collected by the SpamAssassin project for the years 2002-2003 and made available to the research
community. For evaluation in this work, spam messages are not used (only 4150 good messages are used instead).

**PhishingCorpus**: PhishingCorpus contains 4550 phishing emails. These emails were collected by an individual for the period 2004-2007 and donated to the research community. For evaluation, all the phishing emails from this corpus were used.

**Enron Email Dataset**: This dataset contains data from about 150 senior management people of Enron that was made public by the Federal Energy Regulatory Commission during its investigation. This dataset contains approximately 500,000 emails. Out of the Enron emails, we employed 136,226 emails from the inbox and sent folder of the mailbox, thus ensuring only good emails from this corpus.

**SPAM Archive**: SPAM archive contains spam emails collected by Bruce Guenter [147] using various bait accounts since 1998. We used all spam emails of January’2011 though November’2011. This accounted for 336,070 emails thus size of the total corpus was 470,000 (approximately). SPAM archive does not distinguish between ‘spam’ and ‘phishing’ emails. Thus, it is an ideal data set to evaluate the architecture using Co-Training which is a semi-supervised algorithm that employs labeled and unlabeled data.

**PhishTank**: PhishTank URLs are manually verified by human experts that it is a confirmed phish attack. We collected 48,000 phish URLs from phishtank.com for the year 2011.

**Data Preparation**

Two sets of public data set combinations were used to build and evaluate the PLSA model. The first set of experiments (combination1) employed data sets (i) & (ii)
while the second set of experiments (combination2) employed (iii) & (iv). The first set is
a much smaller public corpus than the second set. In combination1, there are a total of
8700 messages, 4550 phishing and 4150 good emails. While in combination1 all emails
are labeled, the combination2, specifically (iv), does not distinguish between phishing
and spam emails. In order to compute misclassification errors, phishing emails in SPAM
archive were segregated using the following semi automated approach. Hyperlinks in
e-mails were extracted using a HTML parser. SURBL [149] provides a reputation lookup
service for domains that are confirmed phish hosting domains. By using a combination of
phishtank.com URLs and domain reputation data from SURBL, if a match is found for
the SURBL domains or phishtank URL in the hyperlinks present in an email, that email is
labeled as a ‘phish’ email. This resulted in phish emails of 47,783 out of 336,070 spam
e-mails. Thus, the distribution of emails in combination2 is 10% phish, 61% spam and
29% good. According to the Internet Security Threat Report’2010 from Symantec [150],
that collected and analyzed billions of emails from 2009, in a realistic mail system 85-
90% of all emails are spam and 5-10% of all spam emails are phish. Thus, to have
realistic distribution of data in combination2, our experiments were conducted with 10%
phish, 80% spam and 10% good emails. Thus, the size of the corpus used for
combination2 is 400,000 emails, which is 10 times the size of corpus used by Bergholz
et. al. [106] and one of the largest email corpus used for phishing detection. Also, we
used public corpus and hence our results can be reproduced.

All the messages were parsed using a MIME parser to separate email headers
from email body. Multipart messages containing HTML parts were further parsed using a
HTML parser to extract the body text and hyperlinks. Both MIME and HTML parsers were written in this work using Java programming language. For evaluation, only messages that contain body text and hyperlinks were considered. Thus messages that failed parser and attachments were not included for building models.

Training and Testing

Experiments were conducted using k-fold cross validation strategy with a k value of 10. Thus, 90% of the dataset was used during training and 10% of the dataset was used for testing. In order to build the PLSA model, the training data is further split into 90% for building the topic model and 10% for computing perplexity. Thus, independent data sets for training, computing perplexity and testing. The TDF matrix builder (see Section 5.3.1) is used to build the term-document matrix for each set. The topic distribution probabilities on the test set is derived using PLSA fold-in (see Chapter 3). Classification in phishGILLNET1 is achieved using Fisher similarity function while phishGILLNET2 employs AdaBoost and phishGILLNET3 employs AdaBoost and Co-Training.

Performance Evaluation Metrics

The performance is measured using the standard measures of performance, namely, Perplexity, Precision, Recall, F-measure and Area under the ROC Curve (AUC) (see section 3.4).

Results - phishGILLNET1

Experiments were conducted using two combinations of data sets, combination1 containing 8K emails and combination2 containing 400K emails. Experiments were
repeated on two machines (i) Mac OS X (10.6.5), 2.66 GHz Intel Core i7, 4GB RAM and (ii) Cent OS (linux 2.6.18), 1.99 GHz Intel Core 2 Duo, 4GB RAM. The average computation time is measured and reported here. For phishing detection, PLSA model consisting of phishing and non phishing topics is first developed. Parsed email data is used to build the term-document-frequency matrix (TDF). After various pre-processing steps, that includes tokenization, stop words removal and porter’s stemming, the part-of-speech (POS) tags are extracted using WordNet [141]. We further observed phishing emails contained intentionally misspelled words, such as, ‘verificacion’, ‘verifcation’, and conjoined words such as ‘yourchasebank’, ‘yourpaypal’. Words that were not found in WordNet direct lookup were further processed using Google’s suggestion API [142] and Levenshtein [143] editing function. If the edit distance is within the threshold value of 5 and if the second lookup in WordNet succeeded, those words were added to build the TDF matrix.

The TEM algorithm, detailed in Section 3.3, is employed to build the PLSA model. The number of topics, $K$, chosen for evaluation includes values ranging from 2 to 200. The maximum number of TEM iterations for convergence was set to 500. The annealing parameter $\beta$ was initialized to value of 1.0 and decremented in increments of 0.25 to see if performance improves on the held out data set.

Results from the PLSA model training and model evaluation are presented in Tables 12, 13 and 14 and Figure 34. In Table 12, the word-to-topic distribution probabilities of top 12 words for two topics (a phishing topic and a non phishing topic) are shown. From this table, it is evident which words make up a phishing topic and which
words make up a good topic. Two methods of evaluating the performance of a PLSA model, log likelihood on the training data and perplexity on the held out, were used. A model that yields lowest log likelihood on the training data is considered the best model. The number of EM steps were varied from 1 to 350. The plot obtained for the (number of topics) $K$ value of 10 topics on combination1 data sets is shown in Figure 34. As it can be seen from the figure, the negative log-likelihood drops steeply until EM iterations of 15 and drops gradually after that indicating (almost) convergence and triggering the stopping criteria. The log likelihood is not a good measure for model generalization. The model with the lowest perplexity is the one that generalizes well for classifying new/unseen data. In order to evaluate performance on held out data, the number of topics was varied from $K=2$ to $K=200$. As it can be seen from Table 13, on the dataset combination1, the perplexity for a $K$ value of 10 yielded 278 and did not change significantly for higher values of $K$. On the dataset combination2, a $K$ value of 200 yielded 1475 and did not change significantly for values larger than 200. The PLSA models were then evaluated for classification performance on test data. This requires computation of topic/document probability distributions on test data. This is achieved using the PLSA’s folding-in technique where the TEM algorithm is employed by keeping the word/topic probability distributions fixed. The Fisher similarity score was then computed between each test data and training data. The label of the training data that yields the highest similarity score is the label of the test data. It can be seen from Table 14 that the PLSA model yielded F-measure of 98.3% on dataset combination1 and 98.1% on dataset combination2. Results on the large public corpus of 400K emails shows the robustness of phishGILLNET1 for
phishing detection. A $K$ value of 200 yielded the best F-measure and lowest false positive on dataset combination2. One can see (Table 14) that performance is almost perfect for $K$ value of 200 and both precision and F-measure are very close to 1. The corresponding computation time (average on two machines) on 200 topic model on dataset combination2 is approximately 3 hours.

Figure 34 phishGILLNET1 Performance - Log Likelihood Versus Number of EM Steps
Table 12 phishGILLNET1 - PLSA Word/Topic Probabilities

| Word (w) | Probability P(w|z) | Topic (z) (phishing) | Word (w) | Probability P(w|z) | Topic (z) (non-phishing) |
|----------|-----------------|----------------------|----------|-----------------|----------------------|
| Bank     | 0.058           |                      | Ocean    | 0.024           |                      |
| Online   | 0.046           |                      | Honolulu | 0.014           |                      |
| Banking  | 0.033           |                      | Imminent | 0.013           |                      |
| America  | 0.032           |                      | Assuring | 0.010           |                      |
| Account  | 0.021           |                      | Handsome | 0.009           |                      |
| Update   | 0.019           |                      | Builder  | 0.007           |                      |
| Security | 0.017           |                      | Lush     | 0.005           |                      |
| Customer | 0.014           |                      | Lousy    | 0.005           |                      |
| Below    | 0.013           |                      | Roads    | 0.005           |                      |
| Link     | 0.013           |                      | Vantage  | 0.005           |                      |
| Click    | 0.011           |                      | Sweetness| 0.005           |                      |
| Please   | 0.011           |                      | Wine     | 0.004           |                      |

Table 13 phishGILLNET1 - PLSA Model Performance

<table>
<thead>
<tr>
<th>Number of topics</th>
<th>Dataset combination 1 (8 K public corpus)</th>
<th>Dataset combination 2 (400 K public corpus)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Perplexity</td>
<td>Computation time (min)</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>2</td>
<td>523.56</td>
<td>1.65</td>
</tr>
<tr>
<td>10</td>
<td>278.81</td>
<td>2.52</td>
</tr>
<tr>
<td>25</td>
<td>277.62</td>
<td>3.12</td>
</tr>
<tr>
<td>50</td>
<td>274.31</td>
<td>3.63</td>
</tr>
<tr>
<td>100</td>
<td>273.33</td>
<td>7.42</td>
</tr>
<tr>
<td>200</td>
<td>271.27</td>
<td>15.38</td>
</tr>
</tbody>
</table>
In order to compare the performance of phishGILLNET1 with that of support vector machines, the TDF matrix of dataset combination2 was utilized. To build the SVM classifier, first the dimensionality reduction technique, Principal Component Analysis, was applied to TDF for computation reasons. In addition, features were selected by applying the information gain criteria. WEKA [86] software using the libSVM library was used to build the SVM classifier. Results from SVM with feature selection are reported in Table 15. It can be seen that SVM results (F-measure of 95.9%) are worse than phishGILLNET1 (F-measure of 98.1%). In addition, SVM took close to 9 hours to train whereas phishGILLNET1 using 200 topics took approximately 3 hours. We report next results of phishGILLNET2 architecture.
Results - phishGILLNET2

Experiments were conducted on the public dataset combination2. The total email corpus of 400,000 emails were used for validating this architecture (40,000 phish, 40,000 good and 320,000 spam). Experiments were conducted using $k$-fold cross validation, with a $k$ value of 10. Thus, for each trial, 90% of the emails were used for training and 10% were used for testing. PLSA topic models were built for number of topics ($K$) 50, 100, and 200. Each model thus results in corresponding number of topic distribution probabilities (features) 50, 100, and 200 respectively. Classifiers were then built using these features and AdaBoost algorithm. Experiments were conducted on two machines (i) Mac OS X (10.6.5), 2.66 GHz Intel Core i7, 4GB RAM and (ii) Cent OS (linux 2.6.18), 1.99 GHz Intel Core 2 Duo, 4GB RAM. The average computation times are measured and reported here. The computation times reported here are the times to perform the cross-validation after extraction of topic features.

Results from the experiments are presented in Tables 16 and 17. Only top five performing classifier results are presented here. The classification performance is reported in Table 16 for 3-class classification and Table 17 for binary classification. For the 3-class problem, boosting with the random forest technique as the base learner yielded the best precision and best F-measure of 97.7% for a $K$ value of 200. For the 2-class problem, boosting using the logistic regression base learner yielded the best precision and F-measure of 99.7% for $k$ value of 200. Thus, for the binary classification phishGILLNET2 resulted in better F-measure (99.7%) compared to phishGILLNET1 (F-
measure 98.1%). Boosting using random forest technique yielded 99.5% for the same number of topics in phishGILLNET2. Random forest is computationally faster than most of the other methods that were evaluated. Results from phishGILLNET2 shows boosting significantly improves classification performance.

<table>
<thead>
<tr>
<th>Table 16 phishGILLNET2 3-Class (Phish/Spam/Good) Classification Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topics</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>200</td>
</tr>
<tr>
<td>200</td>
</tr>
<tr>
<td>200</td>
</tr>
<tr>
<td>200</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 17 phishGILLNET2 2-Class (Phish/Not-Phish) Classification Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topics</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>200</td>
</tr>
<tr>
<td>200</td>
</tr>
<tr>
<td>200</td>
</tr>
<tr>
<td>200</td>
</tr>
<tr>
<td>200</td>
</tr>
</tbody>
</table>
Results - phishGILLNET3

Experiments were conducted using the dataset combination2. Experiments were repeated on two machines to measure the computation time. Of the total corpus of 400K, 10% of the data (40K) was used as labeled data ($L$) for the first iteration of the Co-Training algorithm. The parameters that yielded the best performance in phishGILLNET2 are employed to build phishGILLNET3. This implies that number of topics $K$ is 200 and the weak learner for the AdaBoost is logistic regression (see Table 17). The parameters of the Co-Training algorithm are $p$ (phish)=200 and $n$ (not phish) = 1800. This unbalanced data set is realistic with proportion of phishing emails in large scale mail systems. The size of the unlabeled pool $U'$ is 5000. After each iterations of Co-Training, the text view classifier labels 2000 emails and the hyperlink view labels 2000 emails resulting in 4000 additional labeled data for the next iteration of Co-Training. The pool $U'$ is replenished by selecting 4000 additional emails randomly from the unlabeled set $U$.

Results from the Co-Training algorithm of the combined hyper link and text classifiers are tabulated in Table 18. After 10 iterations of Co-Training, it is evident that phishGILLNET3 results in better performance than phishGILLNET2 (99.8% as compared 99.7%). More iterations of the algorithm resulted in an F-measure 100%. Results show the robustness of PLSA, AdaBoost and Co-Training algorithm to detect phishing. Moreover, phishGILLNET3 achieves superior performance using 10% of the labeled data thus saving time, effort and errors associated with human annotation.
Table 18 phishGILLNET3 Classification Performance

<table>
<thead>
<tr>
<th>Iteration number</th>
<th>TPR</th>
<th>FPR</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
<th>ROC area</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.997</td>
<td>0.014</td>
<td>0.997</td>
<td>0.997</td>
<td>0.997</td>
<td>0.987</td>
</tr>
<tr>
<td>10</td>
<td>0.998</td>
<td>0.015</td>
<td>0.998</td>
<td>0.998</td>
<td>0.998</td>
<td>0.99</td>
</tr>
<tr>
<td>15</td>
<td>0.999</td>
<td>0.014</td>
<td>0.999</td>
<td>0.999</td>
<td>0.999</td>
<td>0.989</td>
</tr>
<tr>
<td>20</td>
<td>1.0</td>
<td>0.012</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>0.991</td>
</tr>
<tr>
<td>25</td>
<td>1.0</td>
<td>0.012</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>0.991</td>
</tr>
<tr>
<td>30</td>
<td>1.0</td>
<td>0.013</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>0.991</td>
</tr>
<tr>
<td>35</td>
<td>1.0</td>
<td>0.015</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>0.991</td>
</tr>
<tr>
<td>40</td>
<td>1.0</td>
<td>0.009</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>0.993</td>
</tr>
<tr>
<td>45</td>
<td>1.0</td>
<td>0.0009</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>0.999</td>
</tr>
</tbody>
</table>

(PLSA 200 topics + AdaBoost with logistic regression weak learner + Co-Training)

Performance Comparison

The performance of phishing detection architecture, phishGILLNET is compared with state-of-the-art research that attempted to solve phishing detection. Performance of each layer of phishGILLNET was compared with ten different published research ranging from year 2007-2011. Comparison was also performed using support vector machines using words (instead of topic probabilities) as features. In Table 19, we show characteristics of our work and state-of-the-art research. The corpus used by phishGILLNET is exclusively public where as in state-of-the-art six of them have used
public, two private and the other two mix of private and public. phishGILLNET has used the largest public corpus of size 400K emails. Thus, results from phishGILLNET are repeatable. The corpus used by phishGILLNET is ten times more than the next (Bergholz et. al. [106]) in terms of size. Thus, phishGILLNET demonstrates the scalability aspect. The most recent public corpus (year 2011) is used by phishGILLNET for evaluation. phishGILLNET2 supports both 3-class (phish, spam, good) and binary (phish, not-phish) classification. The only method that performs 3-class classification is that of Gansterer et. al. [104]. All the others perform binary classification. phishGILLNET3 is the only method that handles unlabeled data. This is the most powerful feature and important contribution of phishGILLNET. To the best of author’s knowledge, there is no other work that applied Co-Training for phishing detection and certainly not at this scale. The closest research work is by Chan et. al. [21] who applied Co-Training for spam classification on a small dataset of 2883 emails.
Results of phishGILLNET comparing state-of-the-art research are tabulated in Tables 20 and 21. The performance metric that is compared is the F-measure (for binary classification) and accuracy (for 3-class classification). On the 3-class classification (see Table 20), the comparison of phishGILLNET2 with the work of Gansterer et. al. [104] on the accuracy metric shows that phishGILLNET2 resulted in a better performance (97.7%) compared with the best result obtained by Gansterer et. al [104]. Thus, topic features using AdaBoost is robust for 3-class classification.
Table 20 Performance Comparison – 3-Class Classification

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy (%)</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>phishGILLNET2</td>
<td>97.70</td>
<td>1</td>
</tr>
<tr>
<td>Gansterer and Pölz [15]</td>
<td>97.00</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 21 Performance Comparison - Binary Classification

<table>
<thead>
<tr>
<th>Method</th>
<th>F-measure (%)</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>phishGILLNET3</td>
<td>100.00</td>
<td>1</td>
</tr>
<tr>
<td>Bergholz et al. [17]</td>
<td>99.89</td>
<td>2</td>
</tr>
<tr>
<td>phishGILLNET2</td>
<td>99.70</td>
<td>3</td>
</tr>
<tr>
<td>Bergholz et al. [16]</td>
<td>99.46</td>
<td>4</td>
</tr>
<tr>
<td>Khonji et al. [20]</td>
<td>99.396</td>
<td>5</td>
</tr>
<tr>
<td>Toolan and Carthy [19]</td>
<td>99.31</td>
<td>6</td>
</tr>
<tr>
<td>phishGILLNET1</td>
<td>98.10</td>
<td>7</td>
</tr>
<tr>
<td>PILFER [10]</td>
<td>97.64</td>
<td>8</td>
</tr>
<tr>
<td>SVM with feature selection (Table 5)</td>
<td>95.90</td>
<td>9</td>
</tr>
<tr>
<td>Abu-Nimeh et al. [12]</td>
<td>NA</td>
<td>-</td>
</tr>
<tr>
<td>Gansterer and Pölz [15]</td>
<td>NA</td>
<td>-</td>
</tr>
<tr>
<td>Al-Momani et al. [21]</td>
<td>NA</td>
<td>-</td>
</tr>
</tbody>
</table>

For the binary classification off the ten state-of-the-art research, only seven of them reported F-measure results. It is evident from the results in Table 21 that phishGILLNET3 resulted in an F-measure of 100%. phishGILLNET3 is the top ranked
method followed by Bergholz et. al. [106], which reported an F-measure of 99.89%. Thus, it is evident that the PLSA, AdaBoost and Co-Training algorithm employed by phishGILLNET3 significantly boosts performance. Moreover, phishGILLNET3 has the additional advantage of not requiring 100% labeled samples thus saving significant manual work. Not relying heavily on manual annotation also has the advantage of the method being less prone to human error and disagreement, as one may consider a spam email as good email and vice versa. Thus, phishGILLNET3 is not only superior on F-measure but also has these additional advantages. phishGILLNET2, that employed AdaBoost classifier and did not employ Co-Training, came close third with a F-measure of 99.70. However, phishGILLNET2 required fully labeled samples unlike phishGILLNET3. Another interesting observation is the top 4 of the top 10 methods employed topic features for building classifiers. While the second and fourth ranked methods utilized features in addition to topic features, phishGILLNET utilized exclusively topic features. Thus, results from the top four methods prove the robustness of using topic features for phishing classification. We present next the phishing website detection methodology.

5.4 Phishing Website Detection

Existing phishing website detection methods are built using features that are susceptible to technology changes. For example, a classifier that uses long Uniform Resource Locator (URL) to distinguish a phishing website will fail for websites hosted at URL shortening services. The content classifiers that uses term-frequency as features do not account for synonyms, words with similar meanings, and, polysemy, words whose
meaning changes according to the context. Moreover, the classifiers were not built for both mobile and desktop clients. According to PayPal [151], sixty seven percent of consumers are expected to use their mobile device for online purchase. The findings by Trusteer [152] conclude that mobile users are three times more vulnerable to phishing attacks than desktop users as mobile devices are always on, users are likely to check messages first on their devices, and devices do not have the same level of protection as desktops. Thus, it is critical for a phishing detection methodology to work not only on desktop clients but also on mobile devices. Furthermore, the past classifier evaluation was limited to English websites.

The main contribution of this research is a content driven phishing website detection methodology that is robust to technology changes, robust to changes in word usage, can be applied to mobile and desktop clients, and is language neutral. The phishing website detection methodology employs the topic model, Latent Dirichlet Allocation (LDA), to extract features, and classification method, AdaBoost, to build the classifier. The phishing website detection methodology is presented next.

5.4.1 Methodology

The methodology developed to detect phishing website is a multi-layered content driven filter that employs the power of natural language processing and machine learning. A schematic representation is shown in Figure 35. The top layer of the detection methodology employs a smart web crawler to capture the contents of phishing and good websites. Attackers who host phishing sites typically employ several methods to prevent detection by automated tools. One of the methods includes multiple redirection of the
phishing URL from one host to the other host. Thus, a phishing detection system may block a phishing URL but the underlying phishing site may continue to live in a different URL for much longer time. The smart web crawler will follow these redirects until the target phishing site found. The other technique that attackers employ is to display a good web page if the crawl is from an automated tool. The smart web crawler employed here creates a user agent for the HTTP request that would mimic normal human initiated request from the web browser. Furthermore, the crawler issues multiple requests to the same URL and compares their contents to ensure the actual content of the site is captured. In order to prevent the smart crawler’s IP address getting blocked by the websites, the requests are sent via an anonymous proxy network service that hides the originating IP address of the web crawler. The proxy network service employed here is the Tor open network [153]. As attackers typically generate dynamic content by employing a scripting language such as javascript, the crawler includes a headless browser that renders the contents of the phishing site. The rendered contents (i.e, contents displayed to the user) are the input to the next layer of the detection methodology.
The second layer of the detection methodology employs a natural language process technique, Latent Dirichlet Allocation (LDA) (see Section 3.3), to discover topics from the rendered web contents. LDA is a topic modeling method, similar to Probabilistic Latent Semantic Analysis (PLSA), that handles synonyms (different words with similar...
meanings) and polysemy (words whose meaning changes according to the context). Since one of the primary goals of this research is to develop a methodology that is robust to changes in technology, the methodology employs rendered web content (as opposed to URL features or HTML features that are susceptible to technology changes) to discover topics using LDA. The LDA topic features are not only robust to changes in technology but also to linguistic variations in the content of the website. The topic distribution probabilities are used as features to build the classifier in the next layer.

The third layer of the detection employs AdaBoost (see Section 3.2) classification method and LDA topic features to build a robust phishing website classifier. AdaBoost is a classifier ensemble method that combines predictions of multiple classifiers to produce a single and robust classifier. The classifier is built using rendered contents on mobile devices and desktop clients. The methodology developed here is robust and language neutral using language translation technique for non-English contents.

5.4.2 Architecture

The architectural components of the phishing detection methodology are presented in this section. A schematic representation of the architecture is shown in Figure 36. The major architectural components are URL fetcher, web crawler, parser, language translator, LDA topic modeler and LDA+AdaBoost classifier.

The URL fetcher has access to good website URLs from two public available websites, DMOZ [154] and Alexa [155]. DMOZ maintains a directory of the web organized into several categories. The URL fetcher fetches website URLs in business, internet, banking, and games, as these categories are the most targeted by attackers. In
addition to DMOZ, the top 500 websites, published by Alexa, were also fetched by the URL fetcher. The good website URLs are downloaded by the fetcher once. Phishing website URLs are fetched from phishtank [148] website every hour of the day. Phishtank [148] provides a dump of confirmed phishing URLs that are online at a given instant. As phishing URLs are short lived, these URLs are fetched periodically. Both good and phishing URLs are stored in a URL database using MySQL.
Figure 36 Phishing Website Detection Architecture
The web crawler fetches the contents of the underlying URLs, both phishing and good ones. Requests from the web crawler are proxy through Tor anonymous network. This prevents a) crawler’s IP address from getting blocked by the website, and, b) to capture the actual contents instead of fake content that the attackers’ website sometime displays. This type of implementation is a unique and novel development compared to the state-of-the-art research on phishing. The generated HTML pages of the website are rendered using the headless browser (i.e., browser with no user interaction). This ensures capturing rendered web contents instead of the raw HTML which sometimes contain nothing more than references to javascript code in the phishing web sites. The good websites requests are also proxy through Tor for latency measurement. The rendered website contents are stored in in a different database on the same mySQL server. Contents are generated and stored for mobile (iPhone, iPAD, Android) and desktop (Windows, Mac) clients.

The parser component parses rendered website contents and extracts hyperlinks, text and images. Images are further converted to text using optical character recognition (OCR) tool. The open source Tesseract [156] was used as the OCR tool. The hyperlinks are converted to text by removing all non alpha-numeric characters. The combined parsed HTML text, hyperlink text and text from image is stored in website text database.

The language translator converts text that are not in English to English. It uses Google’s language detection API [157] to classify the language of the underlying text and calls the translation API [157] for language translation. The translated data is stored in
another database for subsequent processing. This is yet another unique and novel development advanced by the research described here.

The LDA topic modeler builds the topic model from the translated text contents of both phishing and good websites. The LDA model discovers topics and employs Gibbs sampling for parameter estimation. The Stanford topic modeling toolbox [158] is used to implement the topic modeler component. The term document frequency matrix is built after tokenizing the text into words. The standard stop word filter is applied to the tokenized text. The topic modeler outputs topic probability distributions for each document (website) in the corpus. This output is stored in topics database for further processing.

Finally, the classifier is built using LDA topic distributions as input and AdaBoost classification method. Several weak learners are used to build a robust classifier for phishing website detection. The WEKA [86] open source software is used to build the final Adaboost classifier.

5.4.3 Experiments

Details on experiments designed to build and evaluate the phishing website detection methodology are presented in this section. This includes datasets employed, training and testing strategies, metrics to evaluate performance, and results.

Two publicly available good website URL datasets and one publicly available phishing website URL dataset were used to build the phishing detection methodology. Good website URL includes (i) Alex’s top 500 websites [155] and (ii) 52000 URLs from the open directory mozilla project, DMOZ [154] URLs in banking, internet, games and
business categories were selected as they are the most targeted sectors by attackers. For the phishing website URLs, phishtank [148] hourly dump was collected for a period of six months, July’2011 through December’2011. This accounts for 47500 phishing URLs. The rendered website content and translated text of these phishing and good URLs (100,000) are extracted using the architectural components described in the previous section. The top six distribution of phishing content by language were English (68%), Portuguese (13%), French (6%), Spanish (4%), Italian (3%), and German (2%). Overall, there were 34 distinct languages in the phishing content.

Experiments were conducted using k-fold cross validation strategy with a k value of 10. In order to build the LDA model, the training data is further split into 90% for building the topic model and 10% for validating the model’s generalization performance. The topic distribution probabilities obtained from the best topic model were used as features to build the AdaBoost classifier. The quality of the topic model is evaluated using perplexity (see section 3.4). Perplexity, a measure of uncertainty in natural language models, gives a better assessment of how well the model generalizes on unseen data. The lower the perplexity the better the generalization performance. The classification performance is measured using the following standard measures of performance, namely, precision, recall, F-measure, and area under the ROC curve (AUC) (see section 3.4).

Experiments were conducted on a Mac OS X (10.6.5), 2.66 GHz Intel Core i7, 4GB RAM machine. The number of topics to build the LDA topic model was varied from 5 to 300. Experiments were conducted on non-translated desktop client’s rendered
contents, non-translated mobile client’s rendered contents, and translated combined mobile and desktop client’s contents. Results from the LDA topic modeler are shown in Tables 22 and 23. It is evident from Table 22, where selected topics from the 300 topic modeler are listed, that words that make up phishing topics are quite different from that of good topics. From Table 23, it can be seen that the perplexity on the mobile client’s content is lower than that of desktop client’s contents. The perplexity of translated text for the 300 topics LDA model is approximately $1/10^{th}$ of the non-translated text, thus showing the robustness of language translation. The perplexity did not decrease significantly when the number of topics is greater than 300. The best performing 300 topic LDA model for translated text was then used to build the AdaBoost classifier. Results from selected experiments (others omitted due to publication’s page limits) are shown in Table 24. As it can be seen from the results AdaBoost with random forest as the weak learner yielded the best results with an F-measure of 96.3% and an ROC area of 98.9%. The best performing AdaBoost with random forest (RF) weak learner was then further evaluated by varying the proportion of phishing content in training, with the results shown in Table 25. One can see that the best F-measure is 99% when the proportion of phishing in training was 5%.
### Table 22 Phishing Website Detection - LDA Topics

<table>
<thead>
<tr>
<th>Topic (phishing)</th>
<th>Topic (phishing)</th>
<th>Topic (phishing)</th>
<th>Topic (good)</th>
<th>Topic (good)</th>
</tr>
</thead>
<tbody>
<tr>
<td>craigslist</td>
<td>facebook</td>
<td>banking</td>
<td>jazz</td>
<td>web</td>
</tr>
<tr>
<td>ssn</td>
<td>javascript</td>
<td>username</td>
<td>music</td>
<td>hosting</td>
</tr>
<tr>
<td>address</td>
<td>name</td>
<td>forgotten</td>
<td>orchestra</td>
<td>provider</td>
</tr>
<tr>
<td>log</td>
<td>english</td>
<td>safe</td>
<td>band</td>
<td>page</td>
</tr>
<tr>
<td>forgot</td>
<td>birthday</td>
<td>remember</td>
<td>musicians</td>
<td>php</td>
</tr>
<tr>
<td>accounts</td>
<td>maiden</td>
<td>secure</td>
<td>student</td>
<td>internet</td>
</tr>
<tr>
<td>payment</td>
<td>mobile</td>
<td>never</td>
<td>performance</td>
<td>speed</td>
</tr>
<tr>
<td>safety</td>
<td>mother</td>
<td>please</td>
<td>chicago</td>
<td>register</td>
</tr>
</tbody>
</table>

### Table 23 Phishing Website Detection - LDA Model Performance

<table>
<thead>
<tr>
<th># of topics</th>
<th>Desktop client without language translation</th>
<th>Mobile client without language translation</th>
<th>Desktop &amp; mobile clients with language translation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Perplexity</td>
<td>Computation time</td>
<td>Perplexity</td>
</tr>
<tr>
<td>5</td>
<td>5824</td>
<td>1m</td>
<td>5654</td>
</tr>
<tr>
<td>10</td>
<td>5438</td>
<td>3m</td>
<td>5161</td>
</tr>
<tr>
<td>15</td>
<td>5091</td>
<td>4m</td>
<td>5182</td>
</tr>
<tr>
<td>20</td>
<td>5005</td>
<td>6m</td>
<td>4932</td>
</tr>
<tr>
<td>25</td>
<td>4787</td>
<td>8m</td>
<td>4867</td>
</tr>
<tr>
<td>50</td>
<td>4598</td>
<td>18m</td>
<td>4444</td>
</tr>
<tr>
<td>100</td>
<td>4307</td>
<td>40m</td>
<td>4206</td>
</tr>
<tr>
<td>200</td>
<td>4041</td>
<td>1h:30m</td>
<td>3980</td>
</tr>
<tr>
<td>300</td>
<td>3934</td>
<td>2h 38m</td>
<td>3890</td>
</tr>
</tbody>
</table>
### Table 24 Phishing Website Detection - AdaBoost+LDA Classification Performance

<table>
<thead>
<tr>
<th>Topics</th>
<th>Weak learner</th>
<th>TPR</th>
<th>FPR</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
<th>AUC</th>
<th>Computation time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>Deci.Stump</td>
<td>0.918</td>
<td>0.082</td>
<td>0.919</td>
<td>0.918</td>
<td>0.918</td>
<td>0.963</td>
<td>0.94</td>
</tr>
<tr>
<td>30</td>
<td>C4.5</td>
<td>0.959</td>
<td>0.041</td>
<td>0.959</td>
<td>0.959</td>
<td>0.959</td>
<td>0.988</td>
<td>8.27</td>
</tr>
<tr>
<td>30</td>
<td>RIPPER</td>
<td>0.95</td>
<td>0.051</td>
<td>0.95</td>
<td>0.95</td>
<td>0.95</td>
<td>0.984</td>
<td>38.76</td>
</tr>
<tr>
<td>30</td>
<td>RF</td>
<td>0.963</td>
<td>0.037</td>
<td>0.963</td>
<td>0.963</td>
<td>0.963</td>
<td>0.989</td>
<td>15.7</td>
</tr>
<tr>
<td>30</td>
<td>SVM</td>
<td>0.922</td>
<td>0.083</td>
<td>0.922</td>
<td>0.922</td>
<td>0.922</td>
<td>0.962</td>
<td>202.15</td>
</tr>
<tr>
<td>30</td>
<td>Logistic</td>
<td>0.925</td>
<td>0.077</td>
<td>0.925</td>
<td>0.925</td>
<td>0.925</td>
<td>0.958</td>
<td>37.25</td>
</tr>
<tr>
<td>30</td>
<td>RBF</td>
<td>0.918</td>
<td>0.082</td>
<td>0.918</td>
<td>0.918</td>
<td>0.918</td>
<td>0.964</td>
<td>7.04</td>
</tr>
<tr>
<td>30</td>
<td>Naïve Bayes</td>
<td>0.919</td>
<td>0.076</td>
<td>0.921</td>
<td>0.919</td>
<td>0.92</td>
<td>0.957</td>
<td>0.87</td>
</tr>
</tbody>
</table>

### Table 25 Phishing Website Detection - Classification Performance for Varying Phishing in Training

<table>
<thead>
<tr>
<th>% Phishing Training</th>
<th>Weak learner</th>
<th>TP</th>
<th>FP</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
<th>AUC</th>
<th>Computation time</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>RF</td>
<td>0.99</td>
<td>0.19</td>
<td>0.99</td>
<td>0.99</td>
<td>0.99</td>
<td>0.98</td>
<td>5.57s</td>
</tr>
<tr>
<td>10</td>
<td>RF</td>
<td>0.98</td>
<td>0.13</td>
<td>0.98</td>
<td>0.98</td>
<td>0.98</td>
<td>0.97</td>
<td>7.72s</td>
</tr>
<tr>
<td>20</td>
<td>RF</td>
<td>0.97</td>
<td>0.08</td>
<td>0.97</td>
<td>0.97</td>
<td>0.97</td>
<td>0.98</td>
<td>8.23s</td>
</tr>
<tr>
<td>30</td>
<td>RF</td>
<td>0.97</td>
<td>0.06</td>
<td>0.96</td>
<td>0.96</td>
<td>0.96</td>
<td>0.98</td>
<td>11.06s</td>
</tr>
</tbody>
</table>
Performance Comparison

The phishing website detection methodology proposed here is compared with state-of-the-art research for their capabilities, datasets employed and results obtained (see Table 26). The phishing data source used was same for all competing methods. The size of corpus used in our research is 10 times that of state-of-the-art research. Our research is the only one that built models for both mobile and desktop clients and the only one that employed language translation. The 99% TPR was also obtained by Wenyin et al. [159] and Xiang et al. [160]. Xiang et al. [160] used page hash matching to obtain 99% TPR, which is susceptible changes in content. Wenyin et al. [159] method relies on hyperlink relationship to the phishing target in the website content. Thus, websites that collect user credentials but do not have links to phishing targets will fail to be detected by such methods. Our method, which relies on the contents of the website and uses topic features, is robust to both technology changes and changes in word usage. We present next the impersonated entity discovery methodology.
5.5 Impersonated Entity Discovery

The goal of our novel approach is not only to detect phishing attack but also the organization that attacker is impersonating. Towards that goal, we have developed a robust multi-stage content driven methodology, which can be implemented as a filter on email servers and web servers, to automatically detect phishing messages and discover the impersonated entity in those messages. The methodology combines the power of natural language processing and machine learning to build the content filter. The methodology employs named entity extraction and topic discovery methods for feature extraction. Named entities are extracted using Conditional Random Field (CRF) and
topics are discovered using Latent Dirichlet Allocation (LDA). A robust classifier is built by employing topic distribution probabilities and named entities as features and the classification technique, AdaBoost. Once the content is classified as phishing, the methodology employs CRF to identify the impersonated entity, i.e, the organization that this phishing attack is impersonating.

The novel contribution of this research is multi-stage methodology that detects not only phishing but also discovers the impersonated entity from such attacks. While the detection of phishing helps to protect users from falling to identity theft, the automatic discovery of impersonated organization helps legitimate organization to shut down the fake website before their users potentially fall for one. This keeps legitimate company’s customers safe and secure which in turn benefits the company having long lasting customers. It also helps companies to have partnership with other companies to mutually exchange phishing campaign targeted towards each other, which in turn protects respective customers. The combined application of CRF and LDA methods to detect phishing attacks and to discover the impersonated entity is also a novelty of this research. Topic discovery helps to discover impersonated entity and the discovery of impersonated entity helps to discover better topics. The research methodology is presented next.

5.5.1 Methodology

The research methodology employs natural language processing and machine learning to detect phishing attacks and to discover the entity / organization that the attackers impersonate during phishing attacks. The multi-scale methodology first extracts in Stage I, (i) named entities, which includes names of people, organizations, and
locations; and (ii) hidden topics, using CRF and LDA operating on both phishing and non-phishing data. Next in Stage II, utilizing topics and named entities as features, each message is classified as phishing or non-phishing using AdaBoost. Finally in Stage III, for messages classified as phishing, the impersonated entity is discovered using CRF. A schematic representation of the multi-stage research methodology is shown in Figure 37. The multi-stage research methodology and motivations for applying CRF, LDA, and AdaBoost methods are described in this section.

Figure 37 Impersonated Entity Detection Methodology
Stage I – Feature Extraction (CRF, LDA)

The first stage is the feature extraction stage. Two sets of features are extracted during this stage namely, named entities and topics. Named entities are extracted using CRF in Stage I(a) and topics are extracted using LDA in Stage I(b).

Stage I(a) – Named Entity Feature Extraction (CRF)

In this stage, named entities are first extracted, which are then used in Stage II as one set of features to build a classifier for phishing detection. Named entities are proper names (or proper nouns) that are names of people, organizations, locations etc. An example of a “phishing email” and a “non-phishing email” are shown below from the author’s mailbox. Named entities are in bold and italicized.

Phishing Email

“subject: PayPal online : message alert!

Dear Customer

resolution center: your account is limited. regarding this, please follow the link below to resolve this issue:

click here to resolve the problem http://autoplusoman.com/security.html

PayPal - number: id832329-paypal/2011

please allow us 1 to 3 days to resolve your problem. ”

Non-Phishing Email

“Dear Venkatesh Ramanathan,

You just changed your password.
If you didn’t change your password, give us a call right away at 402-935-7733.

Just a reminder:

Never share your password with anyone.

Create passwords that are hard to guess and don’t use personal information.

Be sure to include uppercase and lowercase letters, numbers, and symbols.

Use different passwords for each of your online accounts.

Sincerely,

PayPal

Copyright © 2012 PayPal, Inc. All rights reserved.

PayPal is located at 2211 N. First St., San Jose, CA 95131.”

In the above examples, PayPal is the name of an organization, Venkatesh Ramanathan is the name of a person, San Jose is the name of a city, and CA is the name of a state. In the first stage, we make use of CRF for named entity recognition (NER), which is an information retrieval task that seeks to locate and classify elements in text documents as one of these proper names. CRF is the method of labeling proper names (names of people, locations, organizations, etc.) in a body of text. Given a sentence, the method involves determining words that are named entities and appropriately labeling the entity as the correct proper name. A theoretical description of CRF is given in Chapter 3. A review of the literature reveals that the CRF model has been successfully employed for labeling and parsing sequential data in natural language processing and image processing applications.
Phishing emails typically target financial, social networking, online gaming and email service providers. According to the Anti Phishing Working Group [161], 75% of the phishing attacks target financial institutions, with *PayPal* being one of the top phishing targets. If an email contains the name of a financial institution, such as *PayPal*, there is a high probability that the email is a phishing email. A key motivation for using CRF is its capability to automatically extract named entities from the body of the email, resulting in an improvement to classification accuracy and helping to narrow the search when processing large volumes of emails. It is also possible that the email sent by the financial institution is a legitimate email (see “non-phishing email” example above). The determination whether the email is legitimate or not is made in Stage II using organization names as one of the features.

The second motivation for using CRF is its robustness in extracting named entities (such as names of organizations). CRF extracts names of organizations based on the context in which such words appear, the words that precede the given word, and the words that succeed the given word. In the phishing example above, “is located at” is the context in the following sentence: “*PayPal* is located at 2211 N. First St., *San Jose, CA* 95131.” New organizations could emerge and existing corporations could merge to form a new organization. Attackers may start targeting these new organizations. Since CRF extracts names of companies based on context, we do not have to keep a pre-defined list of all company names.

The third motivation for using CRF is its ability to automatically extract personalized information (names of people) from a given document. Attackers send out
phishing emails in bulk hoping some users would fall for the attack. Thus, most phishing emails are not personalized (see “phishing” example above that has “Dear Customer” whereas “non-phishing” email has “Dear Venkatesh Ramanathan”). This is because: a) the attackers do not have the users’ names, or, b) they know the users’ names but they do not know if the users have accounts with the organization that the attackers are impersonating, or, c) they do not want to spend time in composing millions of personalized emails. Emails from legitimate organizations sent to their registered customers are, in general, personalized as they do have names of the people (see “non-phishing” email example). Thus, automatic extraction of names of the people using CRF, which would normally be present in a good email but usually absent in a phishing email, are employed as additional features for building the classifier. There are several exceptions to this. The targeted phishing attacks, known as spear phishing, where attackers do know user’s names, are personalized. Similarly, legitimate emails such as ones sent from legitimate organizations to potential/prospective customers, respectfully addressed emails (such Dear Sir, Dear Madam), etc., may not be personalized. The presence/absence of personalized information is one of the many features employed by our methodology. Hence, it provides one of the signals (not the whole signal) and also helps to narrow the search to detect phishing attacks.

The final named entity, extracted using CRF, is the location information. The motivation for use of the location information is its presence in a non-phishing email and usually absent in a phishing email. In the non-phishing email sample (shown previously), the location where organization is located, namely, San Jose, CA, is present while it is
absent in the phishing email. It must be noted that it is very easy for an attacker to include
the location information in the phishing email. Thus, it is not as robust a feature as the
other named entities.

Stage I(b) – Topic Feature Extraction (LDA)

Topics comprise the second set of features in the feature extraction stage (see
Figure 37). Group of words or phrases constitute a topic. For example, a “baseball” topic
may comprise words/phrases such as “Yankees”, “home run”, “major league”. A topic
model assigns each document a probability distribution over “topics”, which are in turn a
probability distribution over words/phrases.

The methodology employs LDA to discover topics from a collection of phishing
and non-phishing messages. The motivation for the use of LDA and some significant
differences vis-à-vis PLSA employed in our earlier research [162] for topics discovery is
briefly discussed here, while the theory behind LDA is given in Chapter 3. LDA is a topic
modeling method, similar to PLSA, which employs a bag-of-words approach to discover
hidden theme(s) / topics for given documents. In PLSA, the latent (hidden) topics are
modeled using a weighted likelihood approach instead of using full probability theory,
i.e., PLSA is a non-Bayesian version of LDA. This leads to two problems [27] using
PLSA. First, the number of parameters in the PLSA model grows linearly with the size of
the corpus, which leads to the problem of overfitting. Second, there is no robust method
for the assignment of probabilities to documents outside the training set. LDA overcomes
these problems by defining a generative process for each document wherein the topic distributions are assumed to have a Dirichlet prior.

Here, we apply LDA to discover hidden topics from phishing messages. The motivation for the use of LDA for phishing is illustrated through another sample phishing email from the online banking company CIMB Clicks. Words/phrases that comprise a ‘financial phishing’ topic are shown in bold and italicized.

**Phishing Email**

“**subject: CIMB important notice - account security validation expired**

dear customer

your CIMB Clicks account security validation has expired,

*this maybe as a result of* wrong or *incomplete data*

entered during the *lastupdate.*

it's **strongly required** that you should *validat* your

**bank** account and **confirm** your *internet banking* records.

**click on the following link:**

http://www.ingelam.cl/respaldo/ingelam.php

**fraud prevention unit**

**legal advisor**

**CIMB Clicks security dept. team.”**
One of the motivations for using LDA is its robustness to changes in word usage. It is good at handling synonyms, different words with similar meanings. In a phishing email, attackers seek immediate attention from the user. It is evident from the above example that the ‘financial phishing’ topic comprises the word ‘important’. To seek user attention, attackers may use similar sounding words such as ‘alert’ (see PayPal phishing example), ‘urgent’, etc. Attackers may also choose ‘bank account’ vs. ‘checking account’, ‘rejected’ vs. ‘canceled’, ‘financial institution’ vs. ‘electronic payments association’, etc., and compose a phishing message targeting the same organization. An exact word based filter is not robust to such changes in word usage unlike LDA.

LDA is also robust to polysemy, words with different meaning in different context. For example, a ‘financial phishing’ topic comprises the word “bank” (see above example). The word “bank” in the context of “river bank” is completely a different topic. LDA is robust in discovering those differences.

LDA is robust in discovering the threatening theme in a phishing message that requires the user to act immediately and failure to do so will result in serious repercussions. In the ‘financial phishing’ topic, words/phrases that exhibit such theme include “validation expired”, “incomplete data”, “strongly required”, “confirm”, “click”, etc. This is to convince users to act on those emails by clicking the links contained in the email and filling out the form that require user credentials.

LDA is also robust in discovering topics that contains intentionally misspelled words and conjoined words. The “financial phishing” topic contains misspelt word
‘validat’ and conjoined word ‘lastupdate’. Attackers employ these techniques to avoid detection by exact word based filters. LDA is robust to such tricks by attackers.

The most powerful feature of LDA is its ability to discover multiple topics from a single document. One of the tricks attackers employ in phishing email is to include non-phishing content using white font color HTML attribute. A snippet of such an example is shown below:

```
"<html>
  ...
  subject: important notice
  dear customer,
  we recently reviewed your account, and suspect that your TD Canada Trust online banking account might have been accessed by an unauthorized third party.
  ...
</html>

<text style="FONT-FAMILY: 'Arial','sans-serif';
 COLOR: white; FONT-SIZE: 24pt">
  The candy business underwent a drastic change in the 1830 when technological advances and the availability of sugar opened up the market.
  The new market was not only for the enjoyment of the rich but also for the pleasure of the working class as well. There was also an increasing market
for children. Confectioners were no longer the venue for the wealthy and

... 

</html>”

In the above example, the “financial phishing” topic targeted to TD Canada Trust bank customer is followed by a non-phishing topic, “confectionary”. The non-phishing content (white font on white background) will not be visible to the user’s naked eye. However, this email will pass through filters that treat the document as a whole as most words that are of non-phishing in nature, confectionery in this case, dominate words that are of phishing nature. Since the user will not see the non-phishing topic and the phishing topic sounds legitimate, the user will fall for the phishing attack. However, a server side filter using LDA will discover that there are two distinct topics in the email and assign different probabilities to each topic. Thus, we employ LDA to discover those hidden topics. Once topics are discovered using LDA, the document/topic probability distributions are used as a second set of features for building the phishing classifier.

Stage II – Phishing Classifier (AdaBoost)

The second stage involves building a robust classifier by employing the boosting method, AdaBoost (see Figure 37). Boosting combines many weak and moderately accurate classifiers to build a robust and thus strong classifier for detecting phishing attacks. Boosting also helps to fuse heterogeneous features resulting in improved classification performance. Here, AdaBoost is employed on combined feature sets of
topic distribution probabilities obtained using LDA and named entities obtained using CRF to build a strong classifier for phishing detection. The motivation for combining LDA and CRF methods is as follows. LDA incorporates bag-of-words approach and hence it does not depend on the order of words in the document. CRF relies on preceding and succeeding words to label a word(s) as a particular entity, organization, name or location. While the fusion of LDA and CRF is done at the classification stage using AdaBoost, future work may involve named entity extraction using CRF providing prior probabilities for LDA to yield better and more efficient ways for topics discovery. Likewise, topics discovered using LDA may be used to do build a more focused named entity extraction using CRF.

Stage III – Impersonated Entity Discovery (CRF)

Once the classifier classifies a particular message as phishing, the final stage involves automatic discovery of impersonated entities (see Figure 37). In the phishing examples shown earlier, impersonated organizations include “PayPal”, “CIMB Clicks”, and “TD Canada Trust”. The detection of phishing and subsequent blocking by an email provider helps to protect users from falling prey to identity theft. The automatic discovery of impersonated entities helps legitimate organizations to shut down the fake website before their users potentially fall for one, as their users may be using other email providers that may not have detected and blocked the same phishing messages. This keeps legitimate company’s customers safe and secure, regardless of which systems they use, which in turn benefits the entity / company having long lasting and satisfied
customers. Automatic discovery also helps companies to engage in partnerships with other companies to mutually exchange phishing information. As the impersonating organization discovery is akin to named entity extraction, CRF is employed for automatic discovery of such entities from phishing messages. The generic NER extraction using CRF, employed in the first stage, can extract more than one entity type but is not robust to discover impersonated organizations from phishing messages. Hence, a custom classifier, that is much more focused and specific for impersonated organizations discovery, is built by employing CRF and trained on phishing messages.

The methodology developed in this research not only identifies a phishing message but also identifies the organization that attackers impersonate in the message. The implementation of the corresponding multi-stage architecture is detailed in the next section.

5.5.2 Architecture

The architecture of the developed methodology is shown in Figure 38. The major architectural components are (i) Parser, (ii) CRF Feature Extractor, (iii) LDA Feature Extractor, (iv) AdaBoost Classifier, and, (v) CRF Impersonated Entity Extractor. Each architectural component’s functionality and their sub components will be elaborated below. All the architectural components are implemented using Java programming language with the use of openly available software, when available.

(i) Parser: The Parser component and TDF Matrix Builder sub-component, developed for phishGILLNET (see section 5.3), were employed for this implementation.
(ii) **CRF Feature Extractor**: We employ the Conditional Random Field (CRF) method to extract named entities. The LDA model does not consider named entities, such as names of people, organizations and locations. Since such proper names do not have any linguistic variation, LDA would not be useful to represent them as feature set. CRF is used to extract such named entities from the text of the email using the NER software written by Stanford’s Natural Language Processing Group [163]. The CRF feature extractor includes the following sub components.

![Figure 38 Impersonated Entity Detection Architecture](image)

Figure 38 Impersonated Entity Detection Architecture
CRF Model Loader: Stanford’s NER software comes with a pre-trained model that has been trained on CoNLL, MUC6, MUC7, and ACE datasets. These datasets are the comprehensive source of data containing news articles from United States and United Kingdom. This component loads the pre-trained model in memory to extract named entities from phishing and good emails.

CRF Named Entity Labeler: Given a new document, CRF named entity labeler component identifies and labels each word to most appropriate named entity. Each word is labeled as one of four entities, namely, location, organization, person or other. The labeling is limited to one entity per word. Some words may belong to more than one entity. For example, ‘amazon’ is not only an organization but also a rain forest, ‘Bush’ is a ‘person’, a ‘politician’ and also means a ‘shrub’. The methodology employed here assigns an entity to a word based on the context in which the word appears and the assignment is limited to one entity, which is fine under most circumstances.

CRF Named Entity Extractor: Entities from the labeled document are extracted by this component. These entities serve as one set of features for building the classifier.

(ii) LDA Feature Extractor: This component builds a LDA topic model from input data consisting of phishing and non phishing data and extracts topic/document distribution probabilities for a given new document. These probability distributions serve as second set of features to build the classifier. This component consists of following sub-components:
**TDF Matrix Builder:** This component builds the term-document-matrix using words from the body of the email. This component, developed as part of phishGILLNET implementation (see section 5.3), was reused here.

**LDA Model Trainer:** This component trains and builds a LDA topic model. The input to the model is the TDF matrix of the data set. The trainer employs 90% of the data for building the model and remaining 10% for model’s predictive performance. Stanford’s Topic Modeling Toolbox [158] is employed to implement this component. The LDA trainer uses collapsed variational Bayes approximation algorithm [164], as it is computationally faster and leads to faster model convergence, to build the LDA model. LDA requires number of topics, K, to be specified at initialization similar to cluster analysis. In addition, LDA requires Dirichlet parameters, $\alpha$, parameter of the Dirichlet prior on the per-document topic distributions, and $\beta$, parameter of the Dirichlet prior on the per-topic word distributions, to be specified upfront. The model performance was evaluated by computing perplexity (see Chapter 3 for definition).

**LDA Model Inference:** In model inference, previously trained LDA model is employed to compute probability distributions on the new unseen data set. Here, we use this component to compute per-document topic probability distributions and per-topic word distributions on the test data set. Stanford TMT’s model inference module is used to compute probabilities on the test data set.

**LDA Topic Probability Extractor:** This extracts word/topic and topic/document distribution probabilities computed by the LDA model inference sub-component. Our method makes use of the topic/document distribution probabilities as the second set of
features to build the classifier. By using these probability distributions instead of actual words, the classifier is expected to be robust in detecting phishing attacks.

(iv) AdaBoost Classifier: In order to build a robust phishing classifier we employ the AdaBoost algorithm. The features used to build the classifier are shown in Table 27. The boosting algorithm employs named entities obtained from the CRF model and per-document topic probability distributions obtained from the LDA model. The CRF model extracts named entities using a pre-trained model. Only proper names, locations, organizations and names of people, are included as on set of features. The per-document topic probability distributions of the LDA model that yielded the lowest perplexity are used as second set of features. AdaBoost is robust in building a classifier using these disparate feature sets. The classifier is built using the WEKA software [86].

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P(z_1</td>
<td>d_i)$</td>
<td>Probability that document $d_i$ belongs to topic $z_1$</td>
</tr>
<tr>
<td>$P(z_2</td>
<td>d_i)$</td>
<td>Probability that document $d_i$ belongs to topic $z_2$</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
<td>....</td>
</tr>
<tr>
<td>$P(z_k</td>
<td>d_i)$</td>
<td>Probability that document $d_i$ belongs to topic $z_k$</td>
</tr>
<tr>
<td>Location$_1$</td>
<td>Named entity – location</td>
<td>CRF</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
<td>....</td>
</tr>
<tr>
<td>Location$_L$</td>
<td>Named entity – location</td>
<td>CRF</td>
</tr>
<tr>
<td>Name$_1$</td>
<td>Named entity – name</td>
<td>CRF</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
<td>....</td>
</tr>
<tr>
<td>Name$_N$</td>
<td>Named entity – name</td>
<td>CRF</td>
</tr>
<tr>
<td>Organization$_1$</td>
<td>Named entity – organization</td>
<td>CRF</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
<td>....</td>
</tr>
</tbody>
</table>
(v) CRF Impersonated Entity Extractor: This component extracts organization that the attacker impersonates in the phishing attacks. The component employs CRF to train and build a custom classifier for this purpose. The classifier that comes with Stanford NER is trained on corpus containing news articles that is not robust in extracting impersonating organizations from phishing messages. Hence, we build a custom classifier using phishing corpus specifically to extract impersonated organizations. Most of the core software component of Stanford NER was used to implement this component. Custom components for parsing, tokenizing, labeling and training were implemented in Java. As detailed in the earlier section, automatic extraction of impersonated organizations helps legitimate companies to enforce phishing website take down, which in turn protects their users.

The architecture developed here has two main novel contributions, (i) a phishing detection methodology that combines LDA and CRF methods to build a robust classifier using AdaBoost, and, (ii) impersonated entity extractor using CRF. Experiments conducted on public corpus to evaluate the architecture developed in this research and the architecture’s performance will be reported in the next section.

5.5.3 Experiments

The performance of the multi-stage architecture is evaluated and experimental results are reported here. The architecture is evaluated using openly available standard
data sets containing phishing and non-phishing data. Evaluation of the phishing classifier is done on email data sets. Evaluation of the automatic discovery of impersonated entity is done on phishing emails, phishing URLs and phishing websites.

Datasets and Data Preparation

Three publicly available email datasets were used to evaluate the phishing detection architecture: (i) ham (good) emails from SpamAssassin PublicCorpus [144], (ii) phishing emails from the PhishingCorpus [145], and, (iii) email archive containing spam and phishing emails from SPAM Archive [147]. One publicly available phishing URLs from PhishTank [148] were used for automated email labeling, and, phishing URLs and accompanying phishing websites were used impersonated organizations discovery

SpamAssassin PublicCorpus [144] contains a total of 6047 messages, of which, 4150 messages are good and the remaining are spam. These messages were collected by the SpamAssassin project for the years 2002-2003 and made available to the research community. For evaluation, spam messages were not used (only 4150 good messages are used instead).

The dataset PhishingCorpus [145] contains 4550 phishing emails. These emails were collected by an individual for the period 2004-2007 and donated to the research community. For evaluation, all the phishing emails from this corpus were used.

SPAM Archive [147] contains emails collected by Bruce Guenter using various bait accounts since 1998. For the purpose of evaluation, we employed emails from January’2012 and February’2012. This accounted for approximately 87000 emails. Most emails in SPAM Archive are spam messages. As SPAM Archive does not provide
separate phishing emails, an automated process was performed here to isolate phishing messages from spam messages described as follows.

PhishTank [148] provides confirmed phishing URLs that are verified and labeled as phishing by human experts. Two sets of phishing URLs from PhishTank were created for our experiments. One set included all phishing URLs for the year 2010. This accounted for approximately 196000 URLs. The other set of phishing URLs included August’2011 through February’2012. This accounted for approximately 58000 phishing URLs. The phishing websites corresponding to the second set of phishing URLs, captured using a web crawler were also utilized here for architecture evaluation.

Furthermore, a list of confirmed phishing domains were downloaded from SURBL [149]. Using phishing URLs from the second set and phishing domains from SURBL, emails that contain a phishing URL or a phishing domain in SPAM Archive were separated into a separate phishing email corpus. This accounted for approximately 2200 phishing emails out of the total of 87000 emails in SPAM Archive.

Thus, there is a total of 4550 phishing emails from PhishingCorpus, 4150 good emails from SpamAssassin Corpus, 2200 phishing emails from SPAM Archive and 84800 spam emails from SPAM Archive. These emails were used to validate the phishing classifier. Number of phishing URLs in the dataset are 196000 for the year 2010 and 58000 for the years 2011-2012 and number of phishing websites are 58000 for the years 2011-2012.

All the email messages were parsed using a MIME parser to separate email headers from email body. Multipart messages containing HTML parts were further
parsed using a HTML parser to extract the body text and hyperlinks. For evaluation, only messages that contain either body text or hyperlinks were considered. Thus, messages without message body and messages that failed parser were not included for building models. For building models and validating phishing classifier, phishing emails included all messages from PhishingCorpus and separated phishing emails from SPAM Archive, and, non phishing emails included good emails from SpamAssassin and spam emails from SPAM Archive.

For building the impersonated organizations discovery model using CRF, the phishing data in the data set used included 6750 phishing emails, 254000 phishing URLs and 58000 phishing websites. To build this model and verification using CRF, we needed datasets that has corresponding impersonated organizations identified upfront. Authors manually identified impersonated organization from 3000 phishing emails (2000 from PhishingCorpus and 1000 from SPAM Archive). Some fraction of the phishing URLs from PhishTank have their impersonated organizations pre identified. We considered 50000 such phishing URLs from 2010 and 30000 phishing URLs from 2011-2012 that had impersonated organization in them. The phishing websites included 30000 websites from the year 2011 and 2012. The same phishing email message parsed that was used for evaluating phishing classifier was used for impersonated organizations discovery evaluation. Furthermore, phishing URLs were tokenized by removing non-alpha-numeric characters and phishing websites were parsed using the HTML parser to extract the text portion of website content. The tokenized phishing URLs and parsed phishing websites were used as well for evaluation of the impersonated entity discovery.
Training and Testing

The generic named entity extraction using Stanford NER [163] was performed using the pre-trained model included with the software. This pre-trained model was trained on corpus containing news articles that included CoNLL, MUC6, MUC7, and ACE datasets. The LDA topics discovery model was built by varying the number of topics ‘K’ from 5 to 200. The Dirchlet prior probability distribution parameters \( \alpha \) and \( \beta \) were initialized to 0.1. The maximum number of iterations for convergence was set to 500. The k-fold cross validation strategy was employed to build and evaluate the topics model, with a k value of 10. Thus, the model was built on 90% of the dataset and validated on the remaining 10%. This process was repeated 10 times and the average values are reported here.

The classifier model is built using AdaBoost algorithm. WEKA [86] software is used to build the AdaBoost classifier. The ARFF file, format used by WEKA, is prepared using the combined topic features from LDA and named entity features from CRF. For an email message that does not contain a specific feature, missing value notation was used in the preparation of ARFF file. The k-fold cross validation was used to build the AdaBoost classifier. The number of folds that was used for the experiment was 10. The maximum number of iterations was set to 1000 and the AdaBoost weight threshold was set to 100. The weak learner for AdaBoost was Random Forest algorithm. Experiments were conducted on older (2006) and newer (2012) datasets to evaluate the temporal robustness.
As the pre-trained CRF model is not robust enough to discover impersonated entity (organizations), we built a custom model to perform discovery, using CRF, by employing just phishing messages. Impersonated organizations discovery models were built by using different proportions (50/50 split) of the same data set. Models were also built using different years data to evaluate temporal robustness and different types of data (phishing emails, phishing URLs, phishing websites) to evaluate robustness to data types.

**Performance Measures**

The performance of the LDA topics model is evaluated using perplexity. The perplexity for an LDA model is computed using the equation given in Chapter 3. The model with the lowest perplexity is the one that generalizes well for classifying new/unseen data. The classification performance of phishing classifier is evaluated using the standard measures of performance given in Section 3.4. The performance measure for evaluation of the impersonated organizations discovery model is the fraction of phishing messages in which impersonated organization was correctly discovered.

**Results**

Results from experiments conducted to evaluate the multi-stage architecture are presented in Tables 28, 29, and 30. In Table 28, the performance of the LDA topics model is shown. The topics model was evaluated on older (2006) and newer (2012) to show the temporal robustness of LDA. As it can be seen, the perplexity for a 200 topics model is 232.27 on year 2006 dataset and 873.12 on year 2012 data set. As the perplexity did not reduce significantly when number of topics was increased, the 200 topics model was used to build the phishing classifier. The phishing classifier was built using the
combined feature set, the topics distribution probabilities from LDA and named entities extracted using CRF. The classifier was built using AdaBoost with a Random Forest weak learner. Results from the 10-fold cross validation is presented in Table 3. The AdaBoost classifier was built for varying proportions of phishing email in the data set, 50%, 40%, 30%, 20% and 10%. The classification F-measure for the combined feature set obtained for 50%, 40% and 30% splits were 0.961, 0.987 and 0.988 respectively. The area under the ROC measure (AUC) varied from 0.979 to 1.0 for varying proportions of phishing email in the data set. When the proportion of phishing emails in the data set is less than or equal to 20%, the AdaBoost classifier yields perfect classification (i.e., no misclassification). Thus, it shows the effect of combined topic features and named entities as feature sets and applying boosting in overall performance improvement.

Results were obtained on different year data to show the temporal robustness of the classifier. We conducted experiments on a newer (Jan’2012-Feb’2012) public email corpus that included newer phishing attacks that target social networking sites (such as Facebook) and online gaming sites (such as Sulake) that are not part of the year 2006 dataset. The classification performance in this dataset also yielded F-measure of 100%, thus showing robustness of the phishing classifier.

### Table 28 Impersonated Entity Detection - LDA Performance

<table>
<thead>
<tr>
<th>Data set – 4.5K phishing (PhishingCorpus, 2006), 4.1K non-phishing (SpamAssassin, 2006)</th>
<th>Number of topics</th>
<th>Perplexity</th>
<th>Computation time (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>553.71</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>433.36</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>260.36</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>
### Table 29 Impersonated Entity Detection - Classification Performance

<table>
<thead>
<tr>
<th>% of phishing emails</th>
<th>TPR</th>
<th>FPR</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
<th>Area under ROC (AUC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50%</td>
<td>0.961</td>
<td>0.039</td>
<td>0.961</td>
<td>0.961</td>
<td>0.961</td>
<td>0.979</td>
</tr>
<tr>
<td>40%</td>
<td>0.987</td>
<td>0.013</td>
<td>0.987</td>
<td>0.987</td>
<td>0.987</td>
<td>0.997</td>
</tr>
<tr>
<td>30%</td>
<td>0.988</td>
<td>0.012</td>
<td>0.988</td>
<td>0.988</td>
<td>0.988</td>
<td>0.997</td>
</tr>
<tr>
<td>20%</td>
<td>1.0</td>
<td>0.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>10%</td>
<td>1.0</td>
<td>0.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

### Training/Testing strategy: 10-fold cross validation
Max data: 4.1K phishing, 4.1K non-phishing
Weak learner for AdaBoost: Random Forest

### Table 29 Impersonated Entity Detection - Classification Performance (continued)

<table>
<thead>
<tr>
<th>% of phishing emails</th>
<th>TPR</th>
<th>FPR</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
<th>Area under ROC (AUC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5%</td>
<td>1.0</td>
<td>0.002</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>0.996</td>
</tr>
</tbody>
</table>

### Training/Testing strategy: 10-fold cross validation
Data source: SPAM Archive (2012)
Max data: 2.2K phishing, 84.8K non-phishing
Weak learner for AdaBoost: Random Forest
The results from the impersonated entity discovery model are shown in Table 30.

These results were obtained by training a model using CRF that consisted of only
phishing messages. Models were trained and tested on different data types (phishing emails, phishing URLs, phishing websites) to show the robustness of our approach to disparate data types. In addition, we also show the temporal robustness by training and testing on different years data. As it can be seen from Table 30, the best discovery was achieved when trained and tested on email data sets. The best discovery was 88.1%, i.e., the model was able to discover impersonated organization in 88.1% of the messages that were tested, when trained on 2006 emails and tested on year 2012 emails. Our approach also yielded discovery rate of 76.1% on phishing URLs and 81.6% on phishing websites. The reason for reduced discovery rate on URLs and websites as compared to emails was due to CRF’s reliance on words that precede and succeed and well formed sentences to discover the impersonated entity.

5.6 Summary

This research makes three major contributions to detect phishing attacks: (i) a multi-layered phishing email detection methodology, called phishGILLNET; (ii) a robust phishing website detection methodology; and (iii) impersonated entity discovery.

All three layers of phishGILLNET employ PLSA to discover phishing and non-phishing topics. phishGILLNET1 categorizes unseen data using Fisher similarity. phishGILLNET2 employs AdaBoost using PLSA topic features and builds a better classifier than phishGILLNET1. phishGILLNET3 builds a robust classifier using only a fraction of labeled samples and applying Co-Training to label additional samples. The novelty of this architecture comes from employing semantic features to build the detection model. Intentional misspelled words found in phishing are handled using
Levenshtein editing and Google APIs for correction before building the TDF matrix. One of the important contributions of phishGILLNET is the use of Co-Training on a large corpus of unlabeled data to detect phishing attacks. The architecture developed is compared with ten state-of-the-art methods. The performance of phishGILLNET3 is better than all the other competing methods and achieves a F-measure of 100%.

The multi-layered phishing website detection methodology employs a smart web crawler for capturing rendered website content. The methodology captures contents of desktop clients and mobile devices and applies language translation for content that are not in English. The methodology builds a LDA topic model from the rendered website content. The topic model that yields the best generalization performance is then used to build a robust classifier using AdaBoost. Experimental results show that language translation lowers the perplexity to $1/10^{th}$ of non-translated content. The AdaBoost classifier with random forest as the weak learner provides the best classification performance. The true positive rate and F-measure obtained with 5% phishing content in the training set yielded were 99% which equaled state-of-the-art research. However, our method was evaluated on a much large corpus, it is device neutral and language independent, and hence a significant new research contribution to phishing website detection.

The third major contribution of this research is a multi-stage phishing detection and impersonated entity discovery methodology. CRF is used to extract named entities. Named entities are used as one set of features. Topics are discovered using LDA. The per-document topic probability distributions obtained from the LDA topic model are used
as a second set of features. The combined probability estimates and named entities are used to build a strong classifier using AdaBoost. The 10-fold cross validation is employed to build and validate the phishing classifier. The boosting method resulted in no misclassification on the test set when the percentage of phishing emails is less than 20%. On messages that are classified as phishing, the impersonated entity is discovered by building a model that employs CRF. Impersonated entity is discovered from disparate data types (phishing emails, phishing URLs and phishing websites). Results show that the discovery rate was highest (88.1%) on phishing emails. The discovery rate in phishing URLs was the lowest (76.1%). The unique combined approach that employs CRF and LDA to yield perfect classification for phishing detection, and also enables impersonated entity discovery using CRF, are the novel contributions of this research.
This thesis makes two major contributions towards building a robust identity management system. First, it provides a better and secure authentication mechanism using face recognition for human authentication. Second, it protects people from identity theft using novel and robust content driven phishing detection methodologies.

Two methods for face recognition, namely, ARCF and a hybrid anthropometric and appearance-based approach were developed. ARCF utilizes a recognition-by-parts strategy that employs information that involves both appearance and location. The cluster and strength of the ARCF correlation peaks indicate the confidence of the face authentication made, if any. The development of ARCF, motivated by MACE filters and adaptive beam-forming from radar / sonar, is driven by Tikhonov regularization. The adaptive aspect of ARCF comes from their derivation using both training and test data, similar to transduction, while the robust aspect benefits from the correlation peak optimization to decrease sensitivity to noise and distortions. The comparative advantages of ARCF are motivated, explained, and illustrated vis-à-vis existing correlation filters. Experimental evidence shows the feasibility and reliability of ARCF vis-à-vis occlusion, disguise, and illumination, expression, and temporal variability. The generalization ability of ARCF is illustrated when decision-making thresholds learned a priori from one data base, e.g., FERET, carry over to face images from another data base, e.g., AR.

202
Further improvements will come from using more (gallery and probe) images to learn the ARCF filters for some specific ID as suggested by the FRGC studies carried on images without occlusion and/or disguise.

The complementary hybrid methodology we developed here employs decision-level fusion using backpropagation and feature-level fusion using boosting for the purpose of robust subject authentication vis-à-vis face occlusion and disguise. Holistic anthropometric and appearance-based features feed the data fusion stage. In addition to standard head and face geometric measurements, the proposed holistic anthropometric features extract additional measurements below the face, which describe the neck and shoulder and their contextual relations to head and face. The appearance-based features are derived using PCA or Fisherfaces. Experimental data shows the feasibility and utility of the proposed hybrid (extended geometry + appearance) approach for robust human authentication vis-à-vis occluded and/or degraded face biometrics. The authentication results presented compare favorably against both appearance-based methods and hybrid methods with anthropometric features confined to face and head. The methods proposed can train on clean data and authenticate on corrupt data, or train on corrupt data and authenticate on clean data.

Towards the goal of protecting people from identity theft, this research has developed robust content driven phishing detection methodologies by combining the power of natural language processing and machine learning. A multi-layered methodology, called phishGILLNET, is proposed and evaluated for phishing email detection. All three staged layers of phishGILLNET employ PLSA to discover phishing
and non-phishing topics. phishGILLNET1 categorizes unseen data using Fisher similarity. phishGILLNET2 employs AdaBoost using PLSA topic features and builds a better classifier than phishGILLNET1. phishGILLNET3 builds a robust classifier using only a fraction of labeled samples and applies Co-Training to label additional samples. The novelty of this architecture comes from employing semantic features to build the detection model. Intentional misspelled words found in phishing are handled using Levenshtein editing and Google APIs for correction before building the TDF matrix. One of the important contributions of this methodology is the use of Co-Training on a large corpus of unlabeled data to detect phishing attacks. The architecture developed was compared with ten state-of-the-art methods. The performance of phishGILLNET3 was better than all the other competing methods and achieved a retrieval F-measure of 100%. Evaluation of phishGILLNET3 was done on a very large data set (400K emails) compared to other competing methods. Moreover, the corpus used is publicly available and hence experiments could be reproduced. phishGILLNET3 also has the powerful feature of incorporating unlabeled data during training.

A multi-layered phishing website detection methodology was proposed and evaluated for phishing website detection. The methodology employed a smart web crawler for capturing rendered website content. The methodology captures contents of desktop clients and mobile devices and applies language translation for content that are not in English. The methodology builds a LDA topic model from the rendered website content. The topic model that yields the best generalization performance is then used to build a robust classifier using AdaBoost. Experimental results showed that language
translation lowers the perplexity to 1/10th of non-translated content. The AdaBoost classifier with random forest as the weak learner provided the best classification performance. The true positive rate and F-measure obtained with 5% phishing content in the training set were 99%, which equals state-of-the-art results. However, our method was evaluated on a much large corpus with lower prevalence on phishing, it is device neutral and language independent, and hence it represents a significant new research contribution to phishing website detection.

A robust multi-stage phishing detection and impersonated entity discovery methodology was also developed in this research. CRF was used to extract named entities. Named entities were used as one set of features. Topics were discovered using LDA. The per-document topic probability distributions obtained from the LDA topic model were used as a second set of features. The combined probability estimates and named entities were used to build a strong classifier using AdaBoost. The 10-fold cross validation was employed to build and validate the phishing classifier. The boosting method resulted in no misclassification on the test set when the percentage of phishing emails is less than 20%. On messages that are classified as phishing, the impersonated entity was discovered by building a model that employed CRF. Impersonated entity was discovered from disparate data types (phishing emails, phishing URLs and phishing websites). Results show that the discovery rate was highest (88.1%) on phishing emails. This is due to CRF’s dependence on words that precede, words that succeed, and well-formed sentences to perform robust discovery. The discovery rate in phishing URLs was the lowest (76.1%). The unique combined approach that employed CRF and LDA to
yield perfect classification for phishing detection, and also enabled impersonated entity
discovery using CRF, are the novel contributions of this research. Robust content-driven
phishing detection developed in this research helps service providers to implement this
architecture as a server side filter to eliminate phishing messages before they get to the
user. Upon discovery of an impersonated entity in a phishing message, an organization
can communicate the phishing attack to the entity that is the target of the attack. This in
turn helps that entity to initiate phishing website take down and other counter measures,
thereby protecting customers.

The multi-stage architecture developed in this research extracts named entities
using CRF and discovers topics using LDA, independent of each other. The outputs from
the first stage are then combined using AdaBoost for the classification stage. LDA
incorporates in its model two parameters, $\alpha$, parameter of the Dirichlet prior on the per-
document topic distribution and, $\beta$, parameter of the Dirichlet prior on the per-word topic
distribution. Choosing the right priors helps to discover better topics more efficiently and
minimizes the chance of getting stuck in local optima. One avenue for future research is
to estimate prior probabilities for LDA model using CRF. Knowing the impersonated
entity could also help to discover topics. For example, say PayPal is the impersonated
entity in the majority of the phishing messages. Since PayPal is an online payment
service, the priors can be appropriately chosen to discover online payment phishing
topics from the dataset. LDA does not rely on order of words while CRF does. Thus,
LDA and CRF are complementary to each other with the potential to yield better results.
We consider only messages that we were able to successfully parse using MIME and HTML parser. We also accounted for only text and hyperlinks present in those messages. Some phishing messages could be in image form and some messages, in an adversarial fashion, are constructed to fail parsing. Future work should expand on the architecture advanced here and allow for such messages.

LDA does not consider ordering of words while CRF does consider order. As CRF and LDA are complementary to each other this yields robust results. Furthermore, LDA overcomes the limitations of PLSA employed by phishGILLNET. Results for impersonated entity discovery show that our methodology discovers entities from disparate data sources, phishing emails, phishing URLs and phishing websites. By employing CRF, our methodology is able to discover newer entities (such as Facebook, Sulake) that were not present in the 2006 corpus. Thus, our methodology is robust in impersonated entity discovery, as attackers mostly target entities that are popular and currently successful. These entities could change from time to time. Our methodology is also able to discover variations of a given entity, such as, paypal, paipal, paypal, etc., which attackers employ while creating phishing URLs to resemble a legitimate entity.

Our methodology is domain neutral. It can be employed to detect phishing attacks and discover impersonated entity at social networking posts (Facebook, Twitter, etc.), instant messages, chat, blog posts, etc. As long as the content is available in text, MIME and HTML formats, this architecture can handle all of them.

One of the issues to consider for future evaluation is scaling. A typical email server process billions of message a day. As the goal of this research is to implement this
architecture as a server side filter, further evaluations should be conducted to determine how this architecture scales and whether the classification performance continues to yield no misclassification. One way to evaluate scalability is to implement the architecture using MapReduce framework on a distributed cluster computing platform.

The other issue to consider is how the methodology adapts to changes in phishing attacks. For example, attacks that target social networking and gaming sites were not that prevalent five years ago. Hence, a robust topics model and the resulting phishing classifier must discover these new phishing topics and classify correctly. Our research has shown that we were able to get good results on a smaller but newer 2012 data set. Future research must consider evaluation on much larger data set (millions of emails).

The LDA model requires prior parameters and number of topics pre-specified. A topic discovery model must adapt dynamically to changes in the number of topics. The CRF impersonated entity discovery model must be expanded to adapt to the formation of new entities, including merge and split of existing entities. One approach for implementing such an architecture for a dynamic real-time system would involve autonomic computing. Autonomic computing is an initiative started by IBM that refers to the self-managing characteristics of distributed computing resources, adapting to unpredictable changes while hiding intrinsic complexity to operators and users. It includes self-healing, self-configuration, self-protection and self-optimization. A proposed framework to make the multi-stage architecture autonomic is shown in Figure 39. The performance of the online evaluator and offline modeler was evaluated by implementing the component as a mail filter (MILTER) application [165, 166]. The additional average message processing time
for phishing detection by the online evaluator was 35 milliseconds. Thus, it shows the additional latency introduced by our developed architecture is significantly small.

As classifiers are deployed to detect phishing attacks, adversaries modify their behaviors to avoid detection. The goal of the adversary is to evade detection with minimum effort and cost. An attacker who sends out phishing emails, to get an email past existing email filters, may modify the message by adding or removing words. These changes will make the phishing emails more likely to pass through the filter. Hence,
classifiers must be designed such that they adapt with each new attack and respond automatically to successfully prevent such adversarial attacks. The topic models employed by phishing detection methodologies are robust to such changes in word usage. An attacker may also employ a different strategy by including non-phishing content using white font on a white background in a phishing email. Classifiers that rely on term frequencies will let the phishing email pass through as most words in the email are of non-phishing in nature. Our topic models discover two distinct topics one of phishing and the other of non-phishing in nature in such an email. Hence, rules can be put in place to prevent such emails from getting through the filter. Development of classifiers that are robust to adversarial attacks also require a new framework that combine machine learning and game theory, taking into account the utilities and costs of both the classification system and its adversary.

An integrated biometrics and cybersecurity is another avenue for future research. While biometrics is extremely valuable for robust and reliable authentication, the data contain personally identifiable information that is unique to individuals. If that data lands in the hands of criminals, they could use it to commit fraud, implicate innocent individuals, illegally enter other countries, and gain unauthorized access to various secure facilities. System administrators can set a policy that requires changing password at periodic intervals while biometric is permanent. Hence, it is imperative that biometric data is securely protected using strong encryption methods, and, when lost or stolen the data must be unusable and/or should be self-destructive, i.e., controllable.
Cybersecurity is about securing the infrastructure from attackers. Biometrics not only helps in shielding the infrastructure but also in shielding individuals from identity thefts. Biometrics enhances security of cyberspace in various ways as explained below.

One of the goals of securing cyberspace is to prevent unauthorized access. In order to ensure resources are accessed only by authorized individuals, biometrics provide a superior authentication mechanism. Unlike other authentication mechanisms, as biometrics use biological information that are permanent and unique to individuals, users do not need to use memory based authentication such as passwords or carry around identity cards to prove their identity. As individual’s physical and behavior characteristics are unique, permanent and cannot be forgotten or stolen, biometric technology is an ideal authentication mechanism for controlling secured access.

Biometrics can also be used for person-person authentication in the context of private messaging space. Currently, most messaging (email, IM, etc) does not authenticate/verify if the sender is allowed to send a message to the recipient. Biometrics can help to fill the white space here. For example, one of the ways to secure a unsecure email message is to include biometric data in encrypted form as part of the email message. The data could be decrypted using the key shared by sender and recipient. The public key cryptographic technique could be employed to secure person-person communication. By securing private messaging using biometrics, spear phishing attacks could be thwarted.

Use of biometrics decreases the likelihood of stolen identities. Phishing is about stealing one’s identity. As biological information is harder to steal, it is the ideal solution
to prevent phishing attacks. The password or PIN based authentication require an exact match whereas using biometrics, administrators can set a threshold level to match similarity between what is in the template versus what is being presented. While passwords, PINs and ATM cards can be shared by more than one individual, individual’s biological characteristics cannot be used by another, thus making biometrics a unique and robust authentication mechanism. Thus, biometrics prevents identity theft and helps to build a secure cyberspace.
REFERENCES


annual international ACM SIGIR conference on Research and development in 
information retrieval, vol. 1. (Berkeley, California, USA, 1999), pp. 50–57


model”, Proceedings of the 18th Conference in Uncertainty in Artificial 
Intelligence, 2002.


[30] Lafferty J et al., Conditional random fields: Probabilistic models for segmenting and 
282–289.


[35] H. Wechsler, L. Zimmerman, 2D invariant object recognition using distributed 
associative memories, IEEE Transactions on Pattern analysis and Machine 

[36] H. Wechsler, L. Zimmerman, Distributed associative memory (DAM) for bin- 
picking, IEEE Transactions on Pattern analysis and Machine Intelligence 11 (8) 

International Conference on Automatic Face and Gesture Recognition (FGR), 

[38] R. Everson, L. Sirovich, Karhunen–Loéve procedure for gappy data, Journal of the 


[101] S Abu-Nimeh, D Nappa, X Wang, S Nair, Distributed phishing detection by applying variable selection using Bayesian additive regression trees. in IEEE International Conference on Communications, vol. 1. (Dresden, Germany, 2009), pp. 1–5


CURRICULUM VITAE

Venkatesh Ramanathan received his Bachelor of Engineering (Hons.) in Civil Engineering from Birla Institute of Technology and Science, Pilani, India, in 1990. He received his Master of Science in Civil Engineering from University of Maryland, College Park, Maryland, in 1994. He has over 15 years of experience in architecting and developing software that is scalable, reliable and highly available. His research interests include biometrics, information security, pattern recognition, machine learning and big data analytics.