MULTI-MODAL DECEPTION DETECTION FROM VIDEOS

by
MEHMET UMUT ŞEN

Submitted to
the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

Sabancı University
September 2020

MULTI-MODAL DECEPTION DETECTION FROM VIDEOS

Approved by:

Prof. Berrin Yanıkoğlu .....
(Dissertation Supervisor)

Assoc. Prof. Müjdat Çetin .....

Assist. Prof. Öznur Taştan .....

Assoc. Prof. Erchan Aptoula .....

Assist. Prof. Yakup Genç .....

Date of Approval:

ABSTRACT

MULTI-MODAL DECEPTION DETECTION FROM VIDEOS

MEHMET UMUT ŞEN

Ph.D Dissertation, September 2020

Dissertation Supervisor: Prof. Berrin Yanıkoğlu

Keywords: deception detection, multi-modal, word embeddings, document classification, speech source separation

Hearings of witnesses and defendants play a crucial role when reaching court trial decisions. Given the high-stakes nature of trial outcomes, developing computational models that assist the decision-making process is an important research venue. In this thesis, we address the deception detection in real-life trial videos. Using a dataset consisting of videos collected from concluded public court trials, we explore the use of verbal and non-verbal modalities to build a multimodal deception detection system that aims to classify the defendant in a given video as deceptive or not. Three complementary modalities (visual, acoustic and linguistic) are evaluated separately for the classification of deception. The final classifier is obtained by combining the three modalities via score-level classification, achieving 83.05% accuracy.

Multimodal analysis of trial videos involves many challenges. Prior to developing the final deception detection system, we have worked on sub-problems that would be helpful on improving deception detection performance. High volume of background sounds in a video decreases the quality of the speech features, and it results in low speech recognition performance. We developed a neural network based single-channel source separation model to extricate the speech from the mixed sound recording.

Word embeddings, is the state-of-art technique in processing of textual data. In addition to evaluating pretrained word embeddings in developing the deception system for English, we have also worked on learning word embeddings for Turkish and

used them for categorizing text documents. This work can be applied in future for a deception system in Turkish.

ÖZET

VIDEOLARDAN ÇOKLU-MODALITE ILE ALDATMACA KESTIRIMI

MEHMET UMUT ŞEN

Doktora Tezi, Eylül 2020

Tez Danışmanı: Prof. Berrin Yanıkoğlu

Anahtar Kelimeler: aldatmaca kestirimi, çoklu-modalite, kelime temsilleri, doküman sınıflandırma, konuşma kaynak ayırımı

Sanık ve tanıkların duruşma konuşmaları mahkeme sonuçlarını etkileyen önemli bir faktördür. Mahkeme kararlarının ilgili insanların hayatları üzerinde önemli sonuçlarının olacağı düşünüldüğünde, hakimlerin ve/veya jüri üyelerinin doğru kararları vermelerine yardımcı olabilecek bilgisayımsal modellerin geliştirilmesi önemli bir araştırma alanıdır. Bu tezde, gerçek hayatta geçen mahkeme videolarında aldatmaca saptaması üzerinde çalışılmıştır. Bu amaçla, sonuçlanmış olan kamuya açık mahkemelerin video kayıtlarından oluşan bir verikümesi kullanılmıştır. Verilen bir videodaki kişinin yanıltıcı olup olmadığını kestirmeyi hedefleyen çoklu-modaliteli bir aldatmaca kestirimi sistemi geliştirilmiştir. Aldatmacanın sınıflandırılması için görsel, işitsel ve metinsel olmak üzere 3 farklı modalite ayrı olarak değerlendirilmiştir. Son sınıflandırıcı sistemi, bu 3 farklı modalitenin skor seviyesinde birleştirilmesiyle elde edilmiştir ve 83.05% doğruluk oranıyla aldatmacaları yakalamıştır.

Mahkeme videolarının çoklu-modaliteli analizinin çeşitli zorlukları vardır. Son sistemin geliştirilmesinden önce, aldatmaca kestiriminin performansını artırmaya faydalı olabilecek alt-problemler üzerinde çalışılmıştır. Videolardaki yüksek sesli arkaplan sesleri, konuşma özniteliklerinin kalitesini düşürmektedir; ayrıca otomatik sisteminin içerisinde bulunan konuşma tanıma sisteminin hata oranlarını artırmaktadır. Bu doğrultuda, konuşmaları arka-plan seslerinden ayrıştıran bir yapay sinir ağı temelli tek-kanallı kaynak ayrıştırma modeli geliştirilmiştir.

Kelime temsil vektörleri, metin verisi içeren problemlerin en gelişkin çözümlerinde

kullanılan bir tekniktir. Kelime temsil vektörleri, İngilizce metinsel konuşma kayıtlarından aldatmacanın kestirimi için denenmiş ve iyi sonuçlar alınmıştır. Bunun yanında, kelime temsil vektörlerinin Türkçe üzerindeki başarımının ölçümü üzerine de çalışmalar yapılmış; Türkçe metin kategorizasyonu ve anlambilimsel metin eşleme problemleri için kullanılmıştır. Bu çalışmalar kelime temsil vektörlerinin Türkçe aldatmaca kestirimi probleminde kullanımı için bir ön-çalışma niteliği taşımaktadır.

ACKNOWLEDGEMENTS

I would like to express my deep and sincere gratitude to my thesis supervisor Berrin Yanıkoğlu for her invaluable guidance, tolerance, positiveness, support and encouragement throughout my thesis. I am also grateful to my former thesis superviser Hakan Erdoğan for bringing me into the field and for his guidance and support throughout the earlier years of my doctoral education.

I am grateful to my committee members Müjdat Çetin, Öznur Taştan, Erchan Aptoula and Yakup Genç for taking the time to read and comment on my thesis.

I would like to thank TÜBİTAK for providing the necessary financial support for my doctoral education.

My deepest gratitude goes to my family for their unflagging love and support throughout my life. This dissertation would not have been possible without them.

To my family.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATONS

AU Action Unit ..... xv, 17, 26, 27

AUC Area Under Curve 20, 21, 22, 29, 30

BERT Bidirectional Encoder Representations from Transformers 16, 22

CNN Convolutional Neural Network 30, 71

DNN Deep Neural Network ... 35, 37, 40, 41, 42, 43, 44, 45, 46, 47, 49, 71, 84, 86

LDA Latent Dirichlet Allocation 62, 85, 86

LIWC Linguistic Inquiry and Word Count 3, 4, 16, 22

NLP Natural Language Processing 48, 49, 56, 59, 71, 85, 86

NMF Nonnegative Matrix Factorization 7, 35, 36, 38, 39, 40, 41, 44, 45, 46, 84, 85

NN Neural Network 20, 21, 22, 23, 24, 59, 63, 64, 66, 67, 85

RF Random Forest 20, 21, 78, 80

SDR Signal to Distortion Ratio 46

SIR Signal to Inference Ratio 46

SNR Signal to Noise Ratio 46

SVM Support Vector Machine 20, 29, 59, 62, 63, 64, 65, 66, 78, 80

TF-IDF Term-Frequency Inverse-Document-Frequency . 58, 59, 61, 62, 64, 65, 66

CHAPTER 1

INTRODUCTION

With thousands of trials and verdicts occurring daily in courtrooms around the world, there is a high chance of using deceptive statements and testimonies as evidence. Given the high-stake nature of trial outcomes, implementing accurate and effective computational methods to evaluate the honesty of provided testimonies can offer valuable support during the decision-making process.

The consequences of falsely accusing the innocents and freeing the guilty can be severe. For instance, in the U.S. alone there are tens of thousands of criminal cases filed every year. In 2013, there were 89,936 criminal cases filings in U.S. District Courts and in 2014 the number was 80,262. [1] Moreover, the average number of exonerations per year increased from 3.03 in 1973-1999 to 4.29 between 2000 and 2013. The National Registry of Exonerations reported on 873 exonerations from 1989 to 2012, with a tragedy behind each case (Gross & Warden, 2012). Hence, the need arises for a reliable and efficient system to aid the task of detecting deceptive behavior and discriminate between liars and truth-tellers.

Traditionally, law enforcement entities have made use of the polygraph test as a standard method to identify deceptive behavior. However, this approach becomes impractical in some cases, as it requires the use of skin-contact devices and human expertise to get accurate readings and interpretation. In addition, the final decisions are subject to error and bias not only from the device itself but also from human judgment (Gannon, Beech & Ward, 2009; Vrij, 2001). Furthermore, using proper countermeasures, offenders can deceive these devices as well as human experts.

Given the difficulties associated with the use of polygraph-like methods, machine learning-based approaches have been proposed to address the deception detection

[1] www.uscourts.gov

problem using several modalities, including text (Feng, Banerjee & Choi, 2012) and speech (Hirschberg, Benus, Brenier, Enos, Friedman, Gilman, Gir, Graciarena, Kathol & Michaelis, 2005; Newman, Pennebaker, Berry & Richards, 2003). Unlike the polygraph method, learning-based methods for deception detection rely mainly on data collected from deceivers and truth-tellers. The data is usually elicited from human contributors, in a lab setting or via crowd-sourcing (Mihalcea & Strapparava, 2009; Pavlidis, Eberhardt & Levine, 2002), for instance by asking subjects to narrate stories deceptively and truthfully (Mihalcea & Strapparava, 2009), by performing one-on-one interviews, or by participating in "mock crime" scenarios (Pavlidis et al., 2002).

Despite their potential benefits, an important drawback in data-driven research on deception detection is the lack of real data and the absence of true motivation while eliciting deceptive behavior. Because of the artificial setting, the subjects may not be emotionally aroused or highly motivated to lie, thus making it difficult to generalize findings to real-life scenarios.

In this thesis, we present a multimodal system that detects deception in real-life trial data using verbal, acoustic and visual modalities. The data consists of video clips obtained from real court trials and is initially presented in (Pérez-Rosas, Abouelenien, Mihalcea & Burzo, 2015).

Unlike previous work on this dataset, which focuses on detecting deception at the video-level, we aim to detect deception at the subject-level. We believe this is more in line with the ground-truth for this dataset since it was also obtained at the subject-level: defendants who are found guilty at the end of the trial are labeled as deceptive since they had not admitted to their guilt during the hearings. In the remainder of the thesis, we will refer to this task as a subject-level deception classification.

1.1 Literature Review

Much of the previous study works with the transcriptions of the subjects, i.e. verbal deception detection. Therefore, we split previous work as verbal and non-verbal deception detection and we give summaries of these work at the next sections.

1.1.1 Verbal Deception Detection

Initial work on deception detection focused on statistical methods to identify verbal cues associated with deceptive behavior. Bachenko et al. selected 12 linguistic indicators of deception, including lack of commitment to a statement or declaration, negative expressions, and inconsistencies with respect to verb and noun forms (Bachenko, Fitzpatrick & Schonwetter, 2008). They extracted and analyzed the effect of these indicators on deception for a textual database of criminal statements, police interrogations, depositions and legal testimony. Hauch et al. conducted a meta-study covering 44 studies with a total of 79 linguistic deception cues and obtained a robust analysis of verbal deceptive indicators (Hauch, Blandón-Gitlin, Masip & Sporer, 2015).

To date, works on verbal-based deception detection have explored the identification of deceptive content in a variety of domains, including online dating websites (Guadagno, Okdie & Kruse, 2012; Toma & Hancock, 2010), forums (Joinson & Dietz-Uhler, 2002; Warkentin, Woodworth, Hancock & Cormier, 2010), social networks (Ho & Hollister, 2013), and consumer report websites (Li, Ott, Cardie & Hovy, 2014; Ott, Choi, Cardie & Hancock, 2011). Research findings have shown the effectiveness of features derived from text analysis, which frequently includes basic linguistic representations such as n-grams and sentence count statistics (Mihalcea & Strapparava, 2009), and also more complex linguistic features derived from syntactic CFG trees and part of speech tags (Feng et al., 2012; Xu & Zhao, 2012). Some studies have also incorporated the analysis of psycholinguistics aspects related to the deception process. Some research work has relied on the Linguistic Inquiry and Word Count (LIWC) lexicon (Pennebaker & Francis, 1999) to build deception models using machine learning approaches (Almela, Valencia-García & Cantos, 2012; Mihalcea & Strapparava, 2009) and showed that the use of psycholinguistic information was helpful for the automatic identification of deceit. Following the

hypothesis that deceivers might create less complex sentences to conceal the truth and being able to recall their lies more easily, several researchers have also studied the relation between text syntactic complexity and deception (Yancheva & Rudzicz, 2013).

There is a also significant amount of social science literature that statistically analyzes verbal indicators for deception. Burns et al. extracted LIWC indicators from transcriptions of a set of 911 calls (Burns & Moffitt, 2014). They fed these indicators as features to machine learning classifiers and obtained an accuracy of 84%. Burgoon et al. examined linguistic and acoustic features extracted from a company's quarterly conference call recordings using the Structured Programming for Linguistic Cue Extraction (SPLICE) toolkit (Burgoon, Mayew, Giboney, Elkins, Moffitt, Dorn, Byrd & Spitzley, 2016). They analyzed the strategic and nonstrategic behaviors of deceivers by annotating utterances as prepared (presentation) and unprepared (Q&A) responses and reported significant differences between these two, in terms of deceptive feature statistics. Larcker and Zakolyukina also applied linguistic analysis on conference call recordings from CEOs and CFOs and obtained significantly better deception prediction than a random guess (Bloomfield, 2012; Larcker & Zakolyukina, 2012). Fuller et al. analyzed verbal cues developed by Zhou et al. (Zhou, Burgoon, Nunamaker & Twitchell, 2004; Zhou, Burgoon, Twitchell, Qin & Nunamaker Jr, 2004) and their revised framework using written statements prepared by suspects and victims of crimes on military bases (Fuller, Biros, Burgoon & Nunamaker, 2013). Braun et al. used LIWC indicators to investigate deceptive statements made by politicians labeled by editors of the `politifact.com` website and reported deceptive linguistic indicators in interactive and scripted settings separately (Braun, Van Swol & Vang, 2015).

While most of the data used in related research was collected under controlled settings, only a few works have explored the used of data from real-life scenarios. This can be partially attributed to the difficulty of collecting such data, as well as the challenges associated with verifying the deceptive or truthful nature of real-world data. To our knowledge, there is very little work focusing on real-life high-stake data. The work presented by Vrij and Mann (2001) was the first study, to the best of our knowledge, on a real-life high-stake scenario including police interviews of murder suspects (Vrij & Mann, 2001). Ten Brinke et al. worked on a collection of televised footage from individuals pleading to the public community for the return of a missing relative (ten Brinke & Porter, 2012). The work closest to ours is presented by Fornaciari and Poesio (Fornaciari & Poesio, 2013), which targets the identification of deception in statements issued by witnesses and defendants using a corpus collected from hearings in Italian courts. Following this line of work, we

present a study on deception detection using real-life trial data and explore the use of multiple modalities for this task.

1.1.2 Non-verbal Deception Detection

Earlier approaches to non-verbal deception detection relied on polygraph tests to detect deceptive behavior. These tests are mainly based on physiological features such as heart rate, respiration rate, and skin temperature. Several studies (Derksen, 2012; Gannon et al., 2009; Vrij, 2001) indicated that relying solely on such physiological measurements can be biased and misleading. Chittaranjan et al. (Chittaranjan & Hung, 2010) created audio-visual recordings of the "Are you a Werewolf?" game to detect deceptive behavior using non-verbal audio cues and to predict the subjects' decisions in the game. In order to improve lie detection in criminal-suspect interrogations, Sumriddetchkajorn and Somboonkaew (Sumriddetchkajorn & Somboonkaew, 2011) developed an infrared system to detect lies by using thermal variations in the periorbital area and by deducing the respiration rate from the thermal nostril areas. Granhag and Hartwig (Granhag & Hartwig, 2008) proposed a methodology using psychologically informed mind-reading to evaluate statements from suspects, witnesses, and innocents.

Facial expressions also play a critical role in the identification of deception. Ekman defined micro-expressions as relatively short involuntary expressions, which can be indicative of deceptive behavior (Ekman, 2001). Moreover, these expressions were analyzed using smoothness and asymmetry measurements to further relate them to an act of deceit (Paul, 2003). Ekman and Rosenberg (Ekman & Rosenberg, 2005) developed the Facial Action Coding System (FACS) to taxonomize facial expressions and gestures for emotion- and deceit-related applications. Bartlett et al. (Bartlett, Littlewort, Frank, Lainscsek, Fasel & Movellan, 2006) introduced a real-time system to identify deceptive behavior from facial expressions using FACS. Tian et al. (Tian, Kanade & Cohn, 2005) considered features such as face orientation and facial expression intensity. Owayjan et al. (Owayjan, Kashour, AlHaddad, Fadel & AlSouki, 2012) extracted geometric-based features from facial expressions, and Pfister and Pietikainen (Pfister & Pietikäinen, 2012) developed a micro-expression dataset to identify expressions that are clues for deception. Blob analysis was used to detect deceit by tracking the hand movements of subjects and extracting color features using hierarchical Hidden Markov Model (Lu, Tsechpenakis, Metaxas, Jensen & Kruse, 2005; Tsechpenakis, Metaxas, Adkins, Kruse, Burgoon, Jensen, Meservy,

Twitchell, Deokar & Nunamaker, 2005). Meservy et al. (Meservy, Jensen, Kruse, Twitchell, Tsechpenakis, Burgoon, Metaxas & Nunamaker, 2005) used individual frames as well as videos to extract geometric features related to the hand and head motion to identify deceptive behavior. Caso et al. (Caso, Maricchiolo, Bonaiuto, Vrij & Mann, 2006) identified particular hand gestures that can be related to an act of deception using data collected from simulated interviews including truthful and deceptive responses. Cohen et al. (Cohen, Beattie & Shovelton, 2010) determined that fewer iconic hand gestures were a sign of a deceptive narration using data collected from participants with truthful and deceptive responses. To further analyze the characteristics of hand gestures, a taxonomy of such gestures was developed for multiple applications such as deception and social behaviour (Maricchiolo, Gnisci & Bonaiuto, 2012). Hillman et al. (Hillman, Vrij & Mann, 2012) determined that increased speech prompting gestures were associated with deception while increased rhythmic pulsing gestures were associated with truthful behavior. Vrij and Mann analyzed visual and acoustic features on a dataset of police interviews of murder suspects and reported that convicted subjects "showed more gaze aversion, had longer pauses, spoke more slowly and made more non-ah speech disturbances" when lying than telling the truth (Vrij & Mann, 2001). Ten Brinke et al. manually extracted codings depicting speech, body language and emotional facial expressions for a collection of televised footage in which individuals pleading to the public community for the return of a missing relative (ten Brinke & Porter, 2012). They report informative codings that reflect deception, e.g. liars use fewer words but more tentative words.

Recently, features from different modalities were integrated to find a combination of multimodal features with superior performance (Burgoon, Twitchell, Jensen, Meservy, Adkins, Kruse, Deokar, Tsechpenakis, Lu, Metaxas, Nunamaker & Younger, 2009; Jensen, Meservy, Burgoon & Nunamaker, 2010). An extensive review of approaches for evaluating human credibility using physiological, visual, acoustic, and linguistic features is available in (Nunamaker, Burgoon, Twyman, Proudfoot, Schuetzler & Giboney, 2012). Burgoon et al. (Burgoon et al., 2009) combined verbal and non-verbal features such as speech act profiling, feature mining, and kinetic analysis for improved deception detection rates. Jensen et al. (Jensen et al., 2010) extracted features from acoustic, verbal, and visual modalities following a multimodal approach. Mihalcea and Burzo (Mihalcea & Burzo, 2012) developed a multimodal deception dataset composed of linguistic, thermal, and physiological features. Nunamaker et al. (Nunamaker et al., 2012) provided a review of approaches for evaluating human credibility using physiological, visual, acoustic, and linguistic features. A multimodal deception dataset consisting of linguistic, ther-

mal, and physiological features was introduced in (Pérez-Rosas, Mihalcea, Narvaez & Burzo, 2014), which was then used to develop a multimodal deception detection system that integrated linguistic, thermal, and physiological features from human subjects to create a reliable deception detection system (Abouelenien, Pérez-Rosas, Mihalcea & Burzo, 2014; Abouelenien, Pérez-Rosas, Mihalcea & Burzo, 2016).

1.2 Outline

In Chapter 2, we introduce the deception detection problem in general and the dataset. Then, we present features that we use in our deception detection system and propose new acoustic features. We report results with individual feature sets and their combinations, both with feature concatenation and classifier combination, for the semi-automatic system using some manually labelled features and fully-automatic deception detection systems. Lastly, we analyze the importance of the features and report some cues that are distinctive of deception.

In Chapter 3, we propose a neural network based model for the single-channel source separation problem. After introducing and defining the problem, we explain the traditional nonnegative matrix factorization (NMF) method. Then we define the proposed method which consists of training a deep neural network discriminatevly with individual source utterances and separating mixed test utterances by iteratively minimizing an objective function that includes the outputs of the neural network with respect to the source estimates. We report results of the experiments with a dataset of mixed utterances of piano music and human speech. The work in this chapter can be used in building a deception detection system for videos that includes background sounds.

In Chapter 4, we apply the skip-gram model for learning word embeddings to the Turkish language. After introducing the skip-gram model, we introduce question sets that we produced for measuring the qualities of word embeddings. We conduct experiments with embeddings that are trained with a large Turkish text corpus. We compare hierarchilcal-maximum and negative sampling methods and report that negative-sampling results in better accuracies on almost all cases. We also investigate the effects of embedding dimensions on accuracies and the effect of removing suffixes from the corpus. We finalize the chapter with conclusion and future work. The work in this chapter can be used in building a deception detection system from

videos in Turkish.

After introduction in Chapter 5, we introduce our Turkish document categorization corpora that we downloaded from two news web portals and give descriptive statistics. Then we define document categorization models that we conduct experiments with, including a traditional text classification method of classifying TF-IDF features, neural networks with word embeddings as well as latent dirichlet allocation which is a topic modelling method. Then we define the experimental setup and report the results. The formulation of text categorization problem is the same with detecting deception from lexical modality, therefore we believe this work can be used in building a deception detection system that includes lexical modality.

In Chapter 6, we investigate several text similarity methods for news article matching. After introducing the problem and related work, we define our unsupervised and supervised methods. In the experiments section, we define the dataset, preprocessing steps and report the results.

Finally, in Chapter 7, we give concluding remarks and possible future directions.

1.3 Contributions of the Thesis

Our main contributions in the core part of the thesis are as follows:

- We present a semi-automatic system that can identify deception with 83.05% accuracy using a combination of automatically extracted and manually annotated features, as well as a fully-automatic system that reaches almost 73% accuracy.
- We propose and evaluate new features for the acoustic modality (pitch variations and speech and silence duration histograms), and demonstrated the possibility of using Action Units for automatic visual processing in detecting deception.
- We present insights into the problem by analyzing the importance of features obtained manually and automatically, as well as the linguistic differences among deceptive and truthful subjects.

Furthermore, we have made the following contributions in the related sub-problems:

- We introduce a novel neural network based model for single-channel source separation.
- We trained word embeddings using skip-gram model for Turkish and derived question sets for evaluating word semantic and syntactic linear relationship of word embeddings for Turkish and conduct experiments.
- We collected news articles from two Turkish news portals and experimented for document categorization with various models and report that a neural network with word embeddings outperform other methods.
- We applied Fasttext and Word2vec word embeddings with cosine similarity to the problem of matching news articles from different news sources that have the subjects of the same event and showed that simpler lexical word-counting techniques outperforms word embedding based similarity methods.

CHAPTER 2

MULTIMODAL DECEPTION DETECTION USING REAL-LIFE TRIAL DATA

In this chapter, we introduce the deception detection system developed as the main work of the thesis; describe the experimental setup; and report results on the real-life trial video dataset. We start by introducing the real-life, high-stakes deception dataset. Then, we define the extracted features for different modalities, namely; linguistic, visual and acoustic. We then define the feature integration methods for the subject-level problem setting. Two main deception detection systems are introduced: semi-automatic which includes features that are manually extracted along with features that are automatically extracted; and a fully-automatic system. We report the results and compare them with human performance. Later; we give some insights that are obtained from the models and we compare the results with the state-of-the-art models in the literature at the last section.[1]

2.1 Dataset

We evaluate the developed system using a multimodal deception dataset that is obtained from real-life court trials. The dataset description is included here for completeness; further details can be found in (Pérez-Rosas et al., 2015).

[1] Work at this chapter is published at IEEE Transactions on Affective Computing (Sen, Perez-Rosas, Yanikoglu, Abouelenien, Burzo & Mihalcea, 2020)

2.1.1 Dataset Overview

The dataset consists of trial hearing recordings obtained from public sources. The videos were carefully selected to be of reasonably good audio-visual quality and portray a single subject with his/her face visible during most of the clip duration.

Videos are collected from trials with different outcomes: guilty verdict, non-guilty verdict, and exoneration. For guilty verdicts, deceptive clips are collected from a defendant in a trial and truthful videos are collected from witnesses in the same trial. In some cases, deceptive videos are collected from a suspect denying a crime he/she committed and truthful clips are taken from the same suspect when answering questions concerning some facts that were verified by the police as truthful. For the witnesses, testimonies that were verified by police investigations are labeled as truthful whereas testimonies in favor of a guilty suspect are labeled as deceptive. Exoneration testimonies are collected as truthful statements.

The dataset includes several famous trials (including trials of Jodi Arias, Donna Scrivo, Jamie Hood, and others), police interrogations, and also statements from the "The Innocence Project" website.[2]

2.1.2 Subject-level Ground-truth

In the original dataset, the ground-truth was obtained at video level, by carefully identifying and labeling truthful and deceptive video clips from trial's recordings (Pérez-Rosas et al., 2015).

In this work, we focus on deception at the subject level for two reasons: 1) it is difficult to know the ground-truth of all video clips with certainty and 2) the ultimate goal is to determine whether an individual is being deceptive or not, rather than pinpoint exactly when s/he is lying. Note that subject-level decision is what human jurors are also asked to accomplish during real life trials consisting of several interrogation episodes.

To obtain subject-level ground truth, we only used the trial outcomes to indicate the subject as deceptive or not (deceptive in case of a guilty verdict vs not-deceptive in case of non-guilty verdict or exoneration). The resulting subject-level dataset has

[2] http://www.innocenceproject.org/

Table 2.1 Distribution of gender in the two categories after aggregating individual videos.

	Female	Male	Total
Deceptive	11	13	24
Truthful	12	23	35
Total	23	36	59

Table 2.2 Sample transcripts for deceptive and truthful clips in the dataset.

Truthful	Deceptive
We proceeded to step back into the living room in front of the fireplace while William was sitting in the love seat. And he was still sitting there in shock and so they to repeatedly tell him to get down on the ground. And so now all three of us are face down on the wood floor and they just tell us “don’t look, don’t look" And then they started rummaging through the house to find stuff...	No, no. I did not and I had absolutely nothing to do with her disappearance. And I’m glad that she did. I did. I did. Um and then when Laci disappeared, um, I called her immediately. It wasn’t immediately, it was a couple of days after Laci’s disappearance that I telephoned her and told her the truth. That I was married, that Laci’s disappeared, she didn’t know about it at that point.

59 instances, and the distributions of male vs female and deceptive vs truthful are given in Table 2.1. In the original video-based setting, 45 subjects have single videos, while remaining subjects have a number of videos ranging from 2 to 18. Therefore, aggregation of videos affects 14 of the subjects.

Note that a subject-level deception detection system can be evaluated fairly, by comparing its predictions to the subject-level ground-truth, which is the trial outcome, with the assumption that the trial outcome is correct.

2.1.3 Transcriptions

The transcriptions are obtained using Amazon Mechanical Turk in the original dataset. In video clips where multiple speakers are portrayed (i.e., defendants or witnesses being questioned by attorneys), the AMT workers were asked to transcribe only the subject’s speech, including word repetitions, fillers such as *um*, *ah*, and *uh*, and intentional silences encoded as ellipsis.

The final set of transcriptions consists of 8,055 words, with an average of 66 words per transcript. Table 2.2 shows transcriptions of sample deceptive and truthful

Figure 2.1 Sample screenshots showing facial displays and hand gestures from real-life trial clips. Starting at the top left-hand corner: deceptive trial with forward head movement (*Move forward*), deceptive trial with both hands movement (*Both hands*), deceptive trial with one hand movement (*Single hand*), truthful trial with raised eyebrows (*Eyebrows raising*), deceptive trial with scowl face (*Scowl*), and truthful trial with an up gaze (*Gaze up*).

statements.

2.1.4 Visual Behavior Annotations

Gesture annotations are also available in the dataset.[3] The annotation was conducted using the MUMIN (Allwood, Cerrato, Jokinen, Navarretta & Paggio, 2007) multimodal scheme, which includes several different facial expressions associated with overall facial expressions, eyebrows, eyes and mouth movements, gaze direction, as well as head and hand movements. Sample screenshots showing facial displays and gestures by deceptive and truthful subjects in the dataset are shown in Figure 2.1.

This annotation was done at the video-level by identifying the facial displays and hand gestures that were most frequently observed during the entire clip duration. The annotations are done to simply mark the existence of certain face and hand movements (as binary attributes), due to the time and effort needed to mark the 39 annotations over the course of the video. Two annotators independently labeled a

[3] As done in the Human Computer Interaction Community, "gesture" is used as a broad term that refers to body movements, including facial expressions and hand gestures.

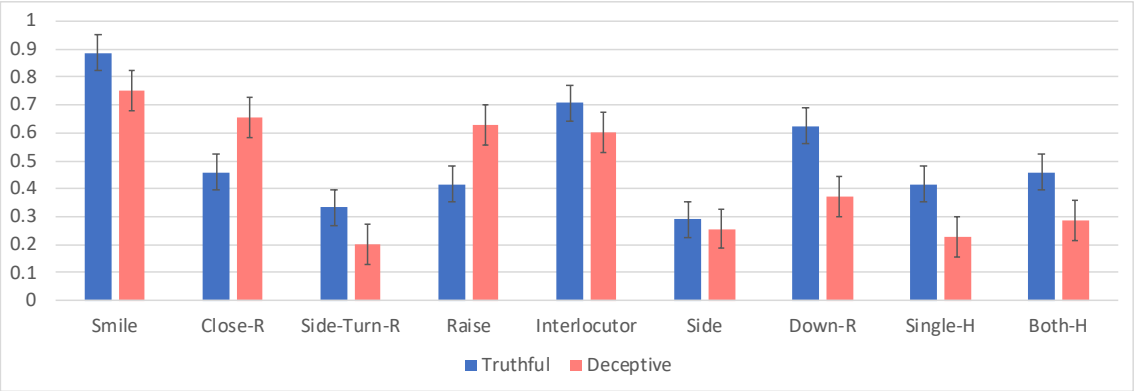


Figure 2.2 Distribution of important visual features for deceptive and truthful groups: Smile, Close-R (closing eyes repeatedly), Side-Turn-R (head turning sides repeatedly), Raise (eyebrow raising), Interlocutor (gazing towards interlocutor), side (gazing to the sides), Down-R (moving the head downwards repeatedly), Single-H (single hand movement), Both-H (moving both hands)

sample of 56 videos. The inter-annotator agreement for this task is shown in Table 2.3. The agreement measure represents the percentage of times the two annotators agreed on the same label for each gesture category. For instance, 80.03% of the time the annotators agreed on the labels assigned to the *Eyebrows* category. On average, the observed agreement was measured at 75.16%, with a Kappa of 0.57 (macro-averaged over the nine categories).

As a preliminary analysis, Figure 2.2 shows the percentages of all the non-verbal features for which we observe noticeable differences for the deceptive and truthful groups. The figure suggests eyebrow (rise) helps differentiate between the deceptive and truthful conditions. Twyman et al. reported that deceivers' right hand moves less during a mock crime experiment (Twyman, Elkins & Burgoon, 2011). This coincides with our single and both hands movement analysis as depicted in Figure 2.2. ten Brinke and Porter (ten Brinke & Porter, 2012) reported that deceptive people blink at a faster rate than genuinely distressed individuals; which also coincides with our findings that deceivers display more frequent occurrence of rapid eye closures, as seen in Fig. 2.). Interestingly, deceivers seem to shake their head (Side-Turn-R) and nod (Down-R) less frequently than truth-tellers while true-tellers seem to move their hands more frequently.

Table 2.3 Gesture annotation agreement

Gesture Category	Agreement	Kappa Score
General Facial Expressions	66.07%	0.328
Eyebrows	80.03%	0.670
Eyes	64.28%	0.465
Gaze	55.35%	0.253
Mouth Openness	78.57%	0.512
Mouth Lips	85.71%	0.690
Head Movements	69.64%	0.569
Hand Movements	94.64%	0.917
Hand Trajectory	82.14%	0.738
Average	75.16%	0.571

2.2 Features for Deception Detection

Aiming to explore the subject-level deception detection with different levels of supervision, we conduct two main experiments using features obtained either manually or semi-automatically. We first present a semi-automatic system where the linguistic and visual feature extraction is done based on manual annotations, as described in Section 2. Second, we build a fully-automatic system that does not rely on human input. Finally, we compare the results with that of human performance on deception detection.

Given the multimodal nature of our dataset, we were interested to evaluate the usefulness of the linguistic, visual, and acoustic components of the recordings, both individually and in combination.

Note that automatic temporal analysis of the videos would be significantly more complicated to accomplish and would require a larger dataset to prevent overfitting; hence it is outside of the scope of this thesis.

The feature extraction process is detailed below.

2.2.1 Linguistic Features

We experimented with linguistic features that have been previously found to correlate with deception cues (Depaulo, Malone, Lindsay, Muhlenbruck, Charlton & Cooper, 2003; Pennebaker & Francis, 1999). These features are derived from the text transcripts of the subjects' statements. In addition, we experimented with word embedding features that map each word to real vector and learned from a large corpus using an unsupervised learning algorithm.

Unigrams We extract unigrams derived from the bag of words representation of each transcript. Each feature consists of frequency counts of unique words in the transcript. For this set, we keep only words with a frequency greater than or equal to 10. The threshold cut was experimentally obtained in a small development set.

LIWC We use features derived from the Linguistic Inquire Word Count (LIWC) lexicon (Pennebaker & Francis, 1999). These features consist of word counts for each of the 80 semantic classes in LIWC. For instance, the class "I" includes words associated with the self (e.g., I, me, myself); "Other" includes words associated with others (e.g., he, she, they); etc.

BERT We use Bidirectional Encoder Representations from Transformers (BERT), which is a language representation model that achieved state-of-the-art results for several language-related problems (Devlin, Chang, Lee & Toutanova, 2018). We used a medium sized BERT model (L=8, H=512) which is pretrained on a large corpus of books and Wikipedia Turc, Chang, Lee & Toutanova (2019). We obtain a vector for each word and average them to get the embedding vector of the utterance.

2.2.2 Annotated Visual Behaviour Features

One set of visual features are derived from the annotations performed using the MUMIN coding scheme described in Section 2.1.4. We create a binary feature for each of the 40 available gesture labels. Each feature indicates the presence of a gesture only if it is observed during the majority of the interaction. The generated features represent nine different gesture categories listed in Table 2.3, covering 32 facial displays and 7 hand gestures.

Facial Displays. These are facial expressions or head movements displayed by the speaker during the deceptive or truthful interaction. They include overall facial expressions such as smiling and scowling; eyebrows, eyes and mouth movements (e.g. repeated eye closing or protruded lips); gaze direction (e.g. looking down or towards the interlocutor); and as well as head movements (e.g. repeated nodding or shaking) and hand movements.

Hand Gestures. The second broad category covers gestures made with the hands, including movements of one or both hands and their trajectories.

2.2.3 Automatically Extracted Visual Features

We automatically extract a second set of visual features consisting of assessments of several facial movements as described below:

Facial Action Units (FACS). These features denote the presence of facial muscle movements that are commonly used for describing and classifying expressions (Ekman, Friesen & Hager, 2002).

We use the OpenFace library (Baltrusaitis, Zadeh, Lim & Morency, 2018) with the default multi-person detection model to obtain 18 binary indicators of Action Units (AUs) for each frame in our videos. These include: AU1 (inner brow raiser), AU2 (outer brow raiser), AU4 (brow lowerer), AU5 (upper lid raiser), AU6 (cheek raiser), AU7 (eyelid tightener), AU9 (nose wrinkler), AU10 (upper lip raiser), AU12 (lip corner puller), AU14 (dimpler), AU15 (lip corner depressor), AU17 (chin raiser), AU20 (lip stretcher), AU23 (lip tightener), AU25 (lips part), AU26 (jaw drop), AU28 (lip suck), and AU45 (blink). We average these binary indicators through the frames and obtain a single AU feature for each video.

2.2.4 Acoustic Features

Previous work has suggested that pitch is an indicator of deceit, and showed that people tend to increase their pitch when they are being deceptive (Streeter, Krauss, Geller, Olson & Apple, 1977). This motivated us to explore whether subjects will

show particular pitch differences in their speech while telling the truth or deceiving.

In addition to pitch, we extracted acoustic features for voiced segments and pauses, based on previous findings showing that deceivers produce slightly shorter utterances and pause more frequently than true-tellers (ten Brinke, Stimson & Carney, 2014). The extracted acoustic features are as follows.

Pitch. We derive features from pitch measurements in the audio portion of each video in the dataset. To estimate pitch, we obtained the fundamental frequency (f0) of the defendants' speech using the STRAIGHT toolbox (Kawahara, Takahashi, Morise & Banno, 2009). Since f_0 is defined only over voiced parts of the speech, we remove unvoiced speech frames from our calculations. We then derive two features (mean and standard deviation) from the raw f_0 measurements: mean$-f_0$ and stdev$-f_0$.

Silence and Speech Histograms. To obtain these features, we run a voice activity detection (VAD) algorithm (Tan & Lindberg, 2010) to obtain the speech and silent segments in the subject's speech. Since the performance of VAD algorithms is affected by the segmentation threshold θ, i.e., high values of θ result on over-segmentation while low values produce under segmentation, we experiment with two values of θ to improve the VAD segmentation in our data: 0.01 and 0.2. After manual inspection, we observed that using a threshold of 0.2, the algorithm segment the audio into words rather than full sentences while a threshold of 0.01 produces full sentence segmentation. Using a VAD threshold of 0.2, with the intent of capturing short pauses, we extract the histograms (using 25 bins) of both voiced and silent segments as features.

Figure 2.3 shows the distribution of the mean and standard deviation of pitch frequencies for the deceptive and truthful groups by gender. As can be seen in this figure, pitch mean values depend on the gender, while standard deviation seems to be more correlated with deception.

Figure 2.4 depicts the histograms of speech and silent lengths by deceptive and truthful subjects. Interestingly, the plot shows that deceptive individuals tend to make shorter pauses more frequently than truthful individuals.

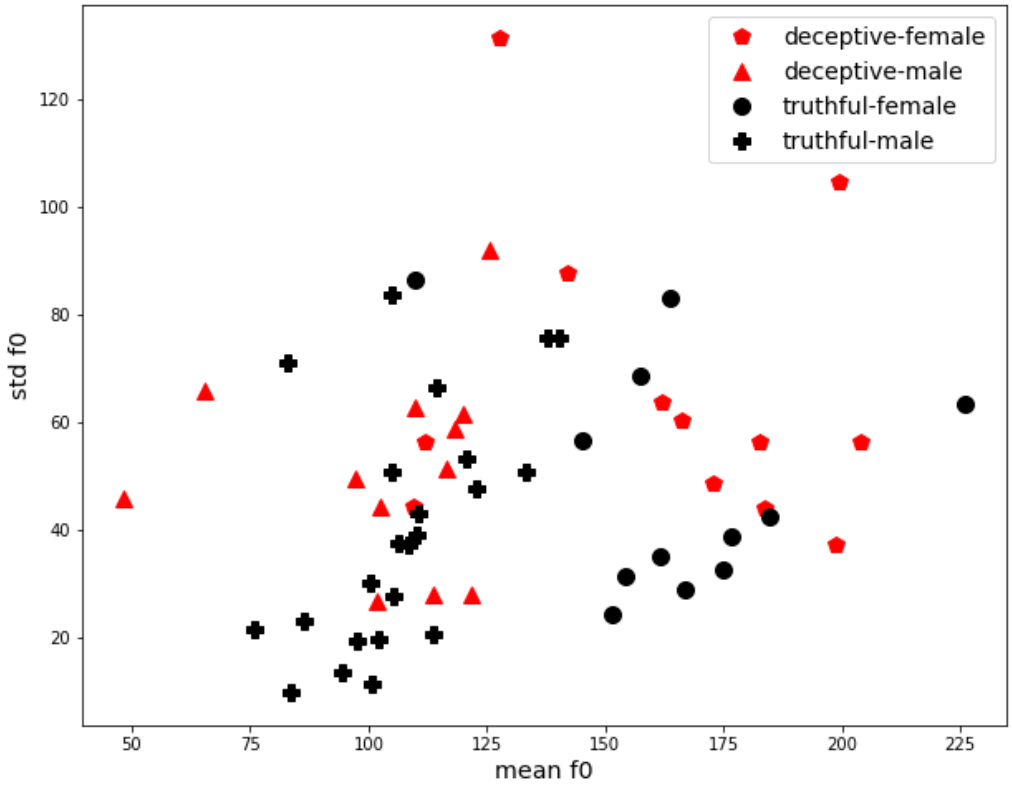


Figure 2.3 Pitch standard deviation vs pitch mean by gender.

2.2.5 Subject-level Feature Integration

Since our feature extraction is performed in each video clip separately for visual features and there are cases where there is more than one video for a single subject, we devised two strategies to aggregate the features across all videos from the same subject. First, taking the maximum values per feature across feature vectors corresponding to every subject's video. Second, averaging the feature values across feature vectors corresponding to each subject's video.

Taking the maximum of the feature values aims to represent single events (e.g., eyes blinking), even if it is observed in just one of the videos belonging to a subject. Averaging the feature values, on the other hand, aims to reduce potential noise introduced during the manual annotation.

During our initial experiments, we found that the averaging strategy outperforms the use of maximum values, hence the former is used during the rest of the experiments reported in the chapter.

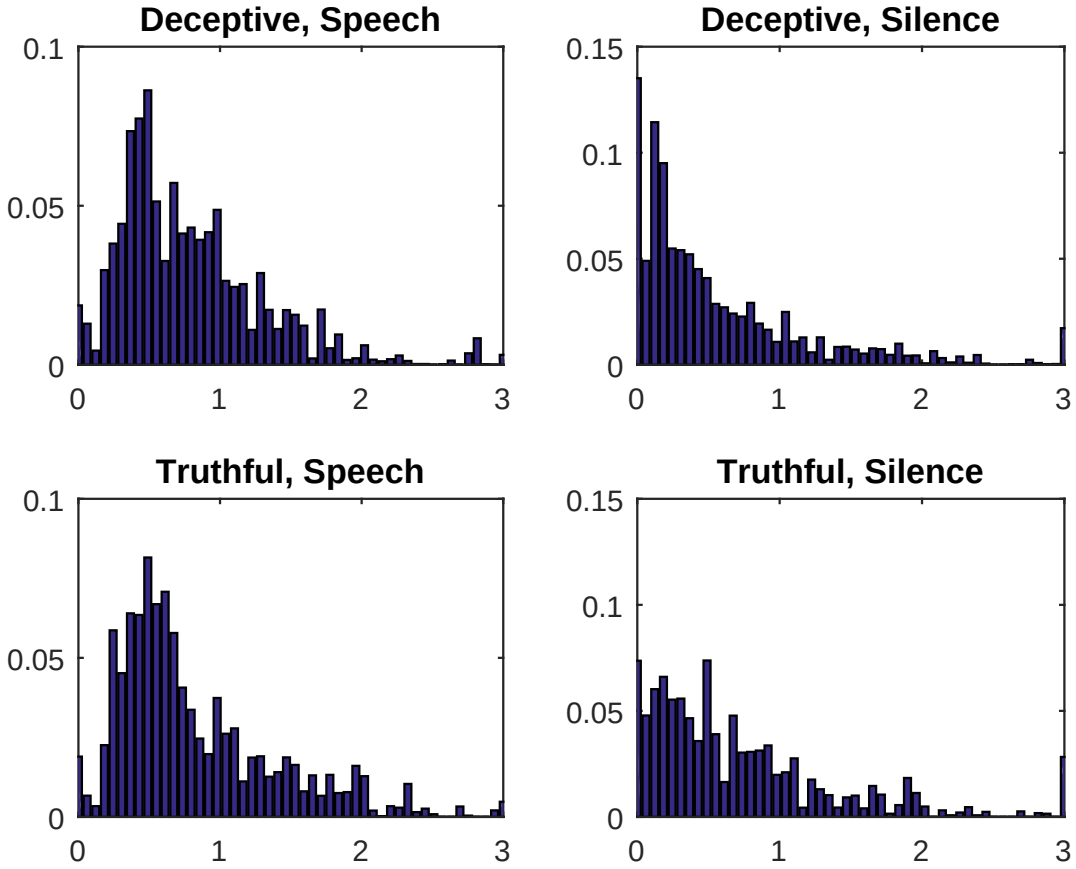


Figure 2.4 Histograms of speech and silence length (measured in seconds) using 25 bins. In all cases, the last bin contains speech or silence segments with duration greater than 3 seconds.

2.3 Classifiers

We chose the Random Forest (RF), Support Vector Machine (SVM) with Radial Basis Function kernel and Neural Network (NN) classifiers, due to their success in many other machine learning problems. For the RF and SVM, we use their implementations as available in Matlab. We use the PyTorch library for the implementation of the NN classifiers (Paszke, Gross, Chintala, Chanan, Yang, DeVito, Lin, Desmaison, Antiga & Lerer, 2017). During our experiments, all classifiers are evaluated using accuracy and area under the curve (AUC) as our main performance metrics.

For the SVM classifiers, we performed parameter tuning over the training set using 4-fold cross-validation separately for each test instance. Specifically, we tune the penalty (C) and the γ parameters of the RBF kernel using grid-search. We applied a 3×3 averaging filter to the resulting loss matrix of the grid search to smooth the parameter tuning results to reduce the noise that results from the low number of data points.

For the RF classifiers, we used the default value for the number of trees (100) and minimum leaf size of 3, without doing parameter optimization.

For the NN classifier, we used a two hidden layers network (100 and 500 nodes for the hidden layers) with ReLu activations along with an output layer with softmax activation function and a cross-entropy loss function. L_2 regularization is applied with a weight of $1E-5$, to prevent over-fitting.

A strong advantage of using RF and the NN classifiers is that they are quite insensitive to the values of their meta-parameters. For instance, when evaluated with different number of hidden nodes in either layer $\{(10,100)$, $(100,100)$, $(500,500)$, $(500,10)$, $(100,10)$, $(10,500)\}$, the NN showed a performance variation of only 1%.

2.4 Semi-Automatic Deception Detection

We develop a semi-automatic system using features derived from manually annotated modalities (visual and linguistic), along with automatically extracted features (speech). Thus, we run several comparative experiments using leave-one-out cross-validation where we test in a single test subject and train in the remaining ones. Furthermore, we run all experiments three times with different random seeds and report the mean and the standard deviation of the results.

2.4.1 Results for Individual Modalities

We initially conduct experiments using each feature set independently and then experiment with different feature combinations using the SVM, RF, and NN classifiers. Table 2.4 shows the results for individual and combined sets of features in each modality.

Among the different classifiers, the RF classifier is the best classifier for most of the linguistic and acoustic features, while the NN performs best with the visual features.

For the visual features, the best results are achieved with the facial displays, reaching an accuracy of 80.79% and an AUC score of 0.94. These results also constitute the

best results across individual feature sets.

For the acoustic features, the best performing feature is the *pitch_stdv*, which represents the standard deviation of the subject's pitch, resulting in an accuracy of 71.19% and an AUC score of 0.79. The rest of the acoustic features obtain significantly lower performance than *pitch_stdv* alone.

For the linguistic modality, the classifier built with the BERT features outperformed unigram features, LIWC features and their combinations. The highest accuracy with lexical features is 68.93% with the Neural Network classifier.

2.4.2 Results for Combined Modalities

For the multi-modal approach, we conduct experiments using two different integration strategies of the three modalities in our dataset: early fusion and late fusion.

2.4.2.1 Early Fusion

First, we experiment with *early fusion* by concatenating the best performing feature sets from the three modalities and using the different classifiers. Results are shown in Table 2.5

During these experiments, the NN classifier consistently obtains the best results among different feature combinations as well as the lowest standard deviation through 3 repetitions of the experiments. Among the different combinations, the combination of features encoding the facial displays, pitch and silence and speech histograms achieve the highest accuracy (83.05%), improving the accuracy obtained with facial display features only by 2.26% points. However, in terms of the AUC, the combination of facial displays and the pitch standard deviation performs the best (0.95).

2.4.2.2 Late Fusion

Second we use *score-level fusion* with classifiers built for individual modalities. For these experiments, we use only the best classifiers and features, leaving out the SVM classifier and hand's gesture features. The aggregated score s_i is obtained as shown in Equation 2.1, where s_{ij} is the score of class c_i obtained with the classifier h_j and w_j is the weight assigned to the classifier h_j.

$$s_i = \sum_j w_j s_{ij} \tag{2.1}$$

We use different classifier weights for the facial displays using increments of 0.1 (the remaining weights are assigned equally to the other classifiers) and report results on the test set. Thus, the best scoring setting is obtained *a posteriori*.

Classification results obtained with this strategy are shown in Table 2.6. We observe that the best result (84.18%) is obtained using the NN classifier and the combination of visual features and acoustic features. This result is higher than the best result obtained with early fusion since it finds the best weights over the test set; but the improvement is very small. The best early fusion results are reported as the proposed system's result, throughout the chapter.

2.5 Fully-Automatic Deception Detection

We also conducted a set of experiments where we explore how well fully automatic feature extraction would work, for our task. Since our acoustic features are already obtained using automatic methods, we focus on the automatic extraction of linguistic and visual features.

We used the OpenFace library (Baltrusaitis et al., 2018) with the default multi-person detection model, to obtain the facial action units (see Section 3.3) for the subject in the video. To address cases where the model identifies multiple persons in the frames, we select the person who is present in the majority of frames as the

person of interest. We manually verified the result of this heuristic and confirmed that in most cases this selection corresponds to the main subject in the video. The software was unable to identify the subject's face in four videos in the dataset, due to the low video quality. These videos are nonetheless included in the evaluation, so as to measure the performance of the system under realistic conditions.

To extract the linguistic features, we applied Automatic Speech Recognition (ASR) to the videos using the Google Cloud Speech API (Google, 2019) and obtained the corresponding transcriptions. Then, as in the manual system, we use these transcriptions to extract lexical features. One shortcoming of the automation here is that the transcriptions also contain the interviewer's speech. Furthermore, the ASR failed to recognize any speech for 10 videos, which correspond to three subjects in the dataset. The obtained transcriptions resulted in an average Word Error Rate (WER) of 0.603 and an insertion rate of 0.152.

The results of the automatic deception system are depicted in Table 2.8. We see that the performance obtained by classifiers build with automatic visual features falls behind the performance obtained when using manual annotations, while automatic extraction of the linguistic features results in a similar performance. As for combined modalities, we see that the best result, 72.88%, (obtained with the fully automatic system, score-level combination, and the NN classifier) is significantly lower than the best performance with the semi-automatic system, 83.05%. However, we would expect the performance gap would to be smaller when using videos that have better visual quality e.g., videos obtained with high-resolution cameras focused on the subject's face.

2.6 Video-Based Deception Detection

We also apply our method to the original dataset with video-level ground-truth labels for completeness. We apply the same experimental setup of leave-one-out scheme with the features and models that resulted in highest accuracies. It should be noted that we remove the other videos of the same person whose video is being tested, from the training data. Accuracies are depicted in Table 2.7. We see that facial displays are again beste features. With the random forest classifier, adding acoustic features to the facial displays increase the accuracy, but adding unigram features results in performance drop.

In general, we obtain lower accuracies with the video-based ground-truth than with the subject-based ground-truth. This result is expected since features in the subject-based setting are extracted from more data and, in addition, when testing a video, we exclude other videos of the subject from the training set to prevent leakage.

2.7 Human Performance

The average human ability to detect deception is reported to be at chance level, while law enforcement professionals can reach 70% (Aamodt & Custer, 2006; Su & Levine, 2016). As part of the work analyzing the importance of multi-modal features in deception detection, Pérez-Rosas et al. (Pérez-Rosas et al., 2015) conducted a study where they evaluate the human ability to identify deceit on trial recordings when exposed to four different modalities: *Text*, consisting of the language transcripts; *Audio*, consisting of the audio track of the clip; *Silent video*, consisting of only the video with muted audio; and *Full video* where audio and video are played simultaneously.

They create an annotation interface that shows instances for each modality in random order to each annotator, and ask him or her to select a label of either "Deception" or "Truth" according to his or her perception of truthfulness or falsehood. The annotators did not have access to any information that would reveal the true label of an instance. The only exception to this could have been the annotators' previous knowledge of some of the public trials in the dataset. A discussion with the annotators after the annotation took place, indicated however that this was not the case.

To avoid annotation bias, they show the modalities in the following order: first they show either *Text* or *Silent video*, then they show *Audio*, followed by *Full video*. Note that apart from this constraint, which is enforced over the four modalities belonging to each video clip, the order in which instances are presented to an annotator is random.

Three annotators labeled all 121 video clips in the dataset, which portray 59 different subjects. To calculate the agreement at the subject-level, they apply majority voting to the labels assigned by each annotator over all the clips belonging to the same subject. They resolve ties by randomly choosing between the deceptive and

truthful labels. Table 2.9 shows the observed agreement and Kappa statistics among the three annotators for each modality.[4] We observe that the agreement for most modalities is rather low and the Kappa scores show mostly poor agreement. As noted before by Ott et al. (Ott et al., 2011), this low agreement can be interpreted as an indication that people are poor judges of deception.

We compare the performance of the three individual annotators and the developed systems, over the four different modalities in the dataset. As shown in Table 2.10, we observe a positive trend in human accuracy in the subject-level deceit detection when using multiple modalities. The trend could be explained by having more deception cues available to them. On average, the poorest accuracy is obtained on *text* only, followed by *Audio*, *Silent video*, and *Full video*, where the annotators have the highest performance. Interestingly, we notice a similar pattern for the developed systems, where we see that having a greater amount of multimodal cues does help to improve the system performance. The fully-automatic system outperforms the average human performance when using each modality individually and in combination (72.88% versus 71.79%). Furthermore, it achieves almost 30% reduction in error compared to the lowest performing human annotator's performance. The semi-automatic system further improves the results of the fully automatic system when using the three modalities (full video), thus suggesting that the feature fusion strategy is also an important aspect when building these models.

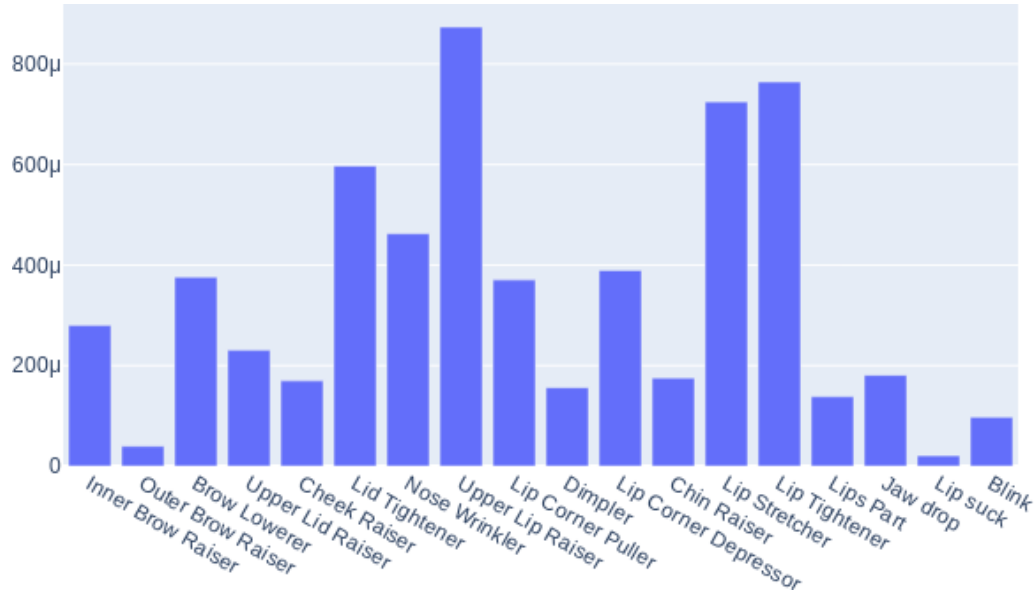


Figure 2.5 Visual feature importance for automatically extracted AU features.

Overall, study of Pérez-Rosas et al. (Pérez-Rosas et al., 2015) indicates that detecting deception is indeed a difficult task for humans and further verifies previous

[4]Inter-rater agreement with multiple raters and variables. `https://nlp-ml.io/jg/software/ira/`

findings where the average human ability to spot liars was found to be slightly better than chance (Aamodt & Custer, 2006). Moreover, the performance of the human annotators appears to be significantly below that of the developed systems.

2.8 Insights for Deception Detection

2.8.1 Visual features

We compute the feature importance scores using the *predictorImportance* function of Matlab (MATLAB, 2010) that bases its estimate on the performance change in the random forest classifier, with the use of each feature. Importance measures of visual AU features are depicted in Figure 2.5. We see that features describing actions of lips reveal substantial deception information (Upper Lip Raiser, Lip Stretcher, Lip Tightener, Lip Corner Depressor, Lip Corner Puller). In addition, (eye)Lid Tightener, Nose Wrinkler, Brow Lowerer and Inner Brow Raiser also have high importance scores.

2.8.2 Deception Language in Trials

To obtain insights into linguistic behaviors displayed by liars during court hearings, we explore patterns in word usage according to their ability to distinguish between the subjects' deceptive and truthful statements. We thus trained a binary Naive Bayes (NB) classifier that discriminates between liars and true-tellers using the unigram features obtained from the subject's statements. We then use the NB model to infer the expected probabilities of each word given its class label. We then sort the words by importance using the following scoring formula:

$$s_i = E[f_i|class = deceptive]/E[f_i|class = truthful], \tag{2.2}$$

In this equation, the expectation E of the word f_i is compared across the deceptive and truthful classes. Note that expectation values are obtained from the resulting NB model rather than empirically from the dataset. The words that are more strongly associated with the deceptive and truthful groups are shown below :

Deceptive Words: not, he, do, 'm, would, his, no, an, mean, with, uh, just, n't, at, but, want, did, if, a, her, any, very, never , ...

Truthful Words: ..., by, so, then, other, was, had, all, through, started, up, on, the, years, two, my, when, of, to, from, um.

In each set, words are shown in decreasing score order i.e., from most deceptive ("not") to most truthful ("um"). We see that negative words such as "not", "no" and "n't" have higher scores, suggesting that deceptive subjects often focus on denying the accusations, whereas truthful subjects are more focused on explaining past events. This coincides with the meta-analysis work of Hauch et al. which shows that deceptive statements have slightly more negative utterances than truthful statements Hauch et al. (2015). Also extreme quantifiers (i.e. "any", "never", "very") occur more frequently in deceptive statements. Houch et al. investigated the effect of certainty on deception and, although certainty indicating words did not have significant effects on deception, they revealed that "deceptive accounts contained slightly fewer tentative words (such as 'may', 'seem', 'perhaps') than truthful accounts" (Hauch et al., 2015). They commented on the possibility of deceivers' motivation to appear credible. Our findings do not coincide exactly, but they are in the same direction.

Newman et al. have found that deceivers have a tendency to use fewer self-referencing expressions, such as "I", "my", "mine" (Newman et al., 2003). This coincides with our findings, because self-referencing words do not appear among the most deceptive words; while the word "my" is one of the most truth-indicating words.

Interestingly, the word "uh" indicates deception whereas the word "um" indicates truthfulness despite both words having the function of pausing.

2.9 Comparison to State-of-Art

Our work extends the work of Pérez-Rosas et al., where the real-life trial dataset was first presented, together with a video-clip level deception detection system (Pérez-Rosas et al., 2015). The main differences with our current work are as follows: i) We conduct the deception detection task at the subject-level rather than at the video-level. Because the outcome of a trial does not say anything about subjects lying in a particular stage of the trial, the subject-level classification better matches the ground-truth labels. To obtain subject-level features, we experiment with different methods for aggregating video-level ground-truth and found that feature averaging across the videos works best. This perspective differentiates our work from all other work conducted on this data set. ii) As a novel contribution, we incorporate the acoustic modality with features designed to detect deception through variations in pitch and silence duration. Our experimental findings indicate that silence duration tends to slightly decrease in deceptive speech and that large pitch variations are strong indicators of deceit. iii) We conducted a linguistic analysis on the differences between deceptive and truthful speech using word-frequency methods. iv) Different from the earlier work, our evaluations are conducted using 3 repetitions for each test sample in the leave-one-subject-out cross-validation, to obtain more robust results. Furthermore, we obtained both accuracy and AUC metrics. v) Finally, our work obtained improved results on the deception detection task (83.05% accuracy and 0.95 AUC with feature-level fusion and 84.18% accuracy and 0.94 AUC with score-level fusion), which are also more reliable due to the cross-validation settings.

Among the studies that report results on this database, Jaiswal et al. used the OpenFace toolkit (Baltrušaitis, Robinson & Morency, 2016) to extract visual features and the OpenSmile toolkit (Eyben, Weninger, Gross & Schuller, 2013) to extract acoustic features which are then fed to an SVM classifier (Jaiswal, Tabibu & Bajpai, 2016). They report a 78.95% accuracy with feature-level fusion, after excluding videos (21 of 121) that are either too short or portray many people such that OpenFace is unable to recognize the subject.

Wu et al. labeled short segments of video clips to train a micro-expression classifier whose outputs are fed to the deception classification system (Wu, Singh, Davis & Subrahmanian, 2017). They report that even though the micro-expression classifier has low performance, its output probabilities are useful to improve the performance of the overall system. They also use GloVe (Global Vectors for Word Representation) embeddings (Pennington, Socher & Manning, 2014) for the linguistic representation

and MFCC features for the acoustic modality. They report an AUC score of 0.92 obtained with a Logistic Regression classifier on a subset of the dataset (104 videos), pruning videos with either significant scene change or human editing.

Krishnamurthy et al. used Convolutional Neural Networks (CNNs) to learn deep representations of the video frames and word embeddings corresponding to the manually annotated transcriptions (Krishnamurthy, Majumder, Poria & Cambria, 2018). These representations are then combined with automatically extracted audio features (obtained using the openSMILE library) and manually annotated visual features in a single hidden layer neural network. The resulting semi-automatic system achieves an AUC score of 0.98 and an accuracy of 96.14%, thus obtaining the best results reported so far on this dataset. However as the authors also acknowledge, there is a possibility that the results may not generalize as well on larger datasets, due to overfitting or learning the idiosyncrasies of the small dataset.

Karimi et al. developed a multimodal deception detection system with automated features (Karimi, Tang & Li, 2018). They employ CNNs followed by a Long-Short Term Memory (LSTM) model to extract the temporal information in the visual and vocal input, along with an attention mechanism focusing on the frames that include visual cues of deception. Their system achieves an accuracy of 84.16% for video-level classification. Venkatesh et al. reported a 97% accuracy on the video-level deception detection using majority voting over the combination of individual modality classifiers (Venkatesh, Ramachandra & Bours, 2019). They used a bag-of-n-grams for lexical features, Cepstral Coefficients for acoustic features, and the visual annotations. However, the last two works appear to have followed a biased experimental protocol: while doing cross-validation, they do not remove other videos of the tested subject from the training set. In that case, the identity of the subject and the labels of the training videos can be used to label the test video. This is also warned against in (Wu et al., 2017).

In summary, existing research on this dataset has approached the problem at the video-level only, obtaining classification performances ranging from 78% to 97%. However, the experimental evaluations are not fully compatible, it is difficult to compare their results directly. For instance, in some work, the videos where the subject is not clearly seen are removed from the dataset; and a subject-based cross validation is not performed in others.

Our results are also not directly comparable with the state-of-art since we detect deception at the subject-level rather than at the video-level. Nonetheless, our best figures obtained with the semi-automatic system (AUC of 0.9462 obtained with a feature-level combination and an AUC of 0.9323 obtained with a score-level combi-

nation of all modalities) are on par with the results of the semi-automatic system of (Krishnamurthy et al., 2018).

2.10 Conclusions

In this chapter, we presented a study of multimodal deception detection using real-life high-stake occurrences of deceit. We use a dataset from public real trials to perform both qualitative and quantitative experiments. We built classifiers relying on the individual or combined sets of verbal and non-verbal features and showed that a system using score-level combination can detect deceptive subjects with an accuracy of 84.18%. Our analysis of non-verbal behaviors occurring in deceptive and truthful videos brought insight into the gestures that play a role in deception. Additional analyses showed the role played by the various feature sets used in the experiments.

We also performed a study of the human ability to detect deception with single or multimodal data streams of real-life trial data. The study revealed high disagreement and low deception detection accuracies among human annotators. Our automatic system using all the modalities outperformed the average non-expert human performance by more than 6% points, and the lowest human annotator's performance by more than 11% points.

In the future, we will work on improving automatic gesture identification and automatic speech transcription, with the goal of taking steps towards a real-time deception detection system.

Table 2.4 Individual feature performance: accuracy (%) and AUC scores. Best results in each line are shown in bold.

Feature Set (dimension)	SVM		RF		NN	
	Accuracy	AUC	Accuracy	AUC	Accuracy	AUC
Visual						
Facial displays (32)	76.27± 0.00	0.8581	76.27 ± 1.69	0.9270	**80.79** ± 0.98	0.9416
Hand gestures (7)	50.28 ± 3.53	0.7232	**64.97** ± 3.91	0.6671	61.58 ± 0.98	0.6930
All visual (39)	58.19 ± 0.98	0.8641	77.40 ± 0.98	0.9187	**78.53** ± 1.96	0.9377
Acoustic						
Pitch (std-f_0) (1)	61.58 ± 0.98	0.6507	**71.19** ± 3.39	0.7939	51.41 ± 0.98	0.7427
Pitch (mean-f_0) (1)	54.24 ± 1.69	0.5223	53.11 ± 0.98	0.5465	**61.02** ± 0.00	0.5235
Sil.Sp.Hist (50)	57.63 ± 0.00	0.4159	**59.32** ± 2.94	0.7069	55.93 ± 1.69	0.6483
All Acoustic (52)	56.50 ± 2.59	0.5864	**63.28** ± 0.98	0.7059	61.02 ± 4.48	0.6589
Linguistic						
Unigrams (134)	53.11 ± 1.96	0.7275	**64.41** ± 4.48	0.6173	63.28 ± 0.98	0.7651
Unigrams - LIWC (100)	52.54 ± 4.48	0.5906	**63.84** ± 2.59	0.6764	55.93 ± 1.69	0.7729
All Unigrams (234)	53.11 ± 4.27	0.6765	**61.58** ± 2.59	0.6605	57.63 ± 1.69	0.7655
BERT (512)	61.58 ± 0.01	0.7482	57.06 ± 0.01	0.6975	**68.93 ± 0.03**	0.8089

Table 2.5 Early fusion results using individual best performing features: accuracy and AUC scores. Best results are shown in bold.

Modalities	SVM		RF		NN	
	Accuracy	AUC	Accuracy	AUC	Accuracy	AUC
Facial Displays	76.27±0.00	0.8581	76.84 ± 0.80	0.9270	**80.79** ± 0.98	0.9416
+ Pitch (std-f_0)	53.11 ± 5.45	0.7148	62.71 ± 1.69	0.6511	**82.49** ± 0.98	**0.9462**
+ Pitch (std-f_0) + Sil.Sp.Hist.	68.93 ± 0.98	0.8585	66.10 ± 11.86	0.8649	**83.05** ± 1.69	0.9166
+ All Acoustic	72.32 ± 0.98	0.8604	67.80 ± 2.94	0.8482	**82.49** ± 0.98	0.9153
+ LIWC	54.24 ± 1.69	0.8173	63.84 ± 1.96	0.7778	**75.14** ± 1.96	0.8961
+ Unigrams	55.93 ± 3.39	0.7928	63.28 ± 0.98	0.7310	**78.53** ± 0.98	0.8903
+ Pitch (std-f_0) + Sil.Sp.Hist. + LIWC	53.67 ± 3.53	0.8281	65.54 ± 1.96	0.7852	**75.14** ± 1.96	0.8780
+ BERT	62.15± 2.6	0.8080	58.19 ± 2.6	0.7182	**71.75** ± 0.98	0.8831
+ All Acoustic + Unigrams	57.06 ± 0.98	0.8072	63.84 ± 3.53	0.7494	**75.71** ± 0.98	0.8778

Table 2.6 Late fusion results using best performing features and different classifier weight combinations. Face refers to facial displays and pitch refers to std-f_0. The results are obtained *a posteriori* and best results are shown in bold.

Visual weight	RF			NN		
	+Acoustic	+Linguistic	+Acoustic +Linguistic	+Acoustic	+Linguistic	+Acoustic +Linguistic
$w_{face} = 1.0$	76.84 ± 0.80			80.79 ± 0.98		
$w_{face} = 0.9$	77.40 ± 0.98	77.97 ± 1.69	77.40 ± 0.98	81.92 ± 0.98	83.05 ± 0.00	81.92 ± 0.98
$w_{face} = 0.8$	77.40 ± 2.59	77.97 ± 1.69	77.40 ± 0.98	83.62 ± 1.96	79.66 ± 0.00	83.05 ± 0.00
$w_{face} = 0.7$	79.10 ± 2.59	76.84 ± 0.98	78.53 ± 1.96	**84.18** ± 0.98	80.79 ± 0.98	81.92 ± 1.96
$w_{face} = 0.6$	76.84 ± 0.98	76.27 ± 0.00	76.84 ± 1.96	**84.18** ± 1.96	79.66 ± 1.69	82.49 ± 0.98
$w_{face} = 0.5$	76.27 ± 1.69	68.93 ± 2.59	74.01 ± 5.18	81.92 ± 0.98	78.53 ± 0.98	83.62 ± 1.96

Table 2.7 Results for video-based setting

Modalities	SVM		RF		NN	
	Accuracy	AUC	Accuracy	AUC	Accuracy	AUC
Facial Displays	**72.73** ± 0.00	0.6005	69.15 ± 0.95	0.6680	68.60 ± 2.19	0.6734
+ All Acoustic	64.74 ± 2.08	0.6530	69.97 ± 0.48	0.6997	68.59 ± 0.00	0.6901
+ Unigrams	61.43 ± 1.26	0.5490	51.52 ± 1.72	0.4191	64.46 ± 1.43	0.6288
+ All Acoustic + Unigrams	59.78 ± 2.66	0.5472	55.65 ± 1.26	0.4469	66.39 ± 2.08	0.6372

Table 2.8 Fully-automatic system: classification accuracies with individual (top 3 rows) and combined modalities (bottom 2 rows)

Modality	SVM	RF	NN
Visual (Action Units)	53.67%	61.58%	57.63%
All Acoustic	56.50%	63.28%	61.02%
Linguistic (Unigrams)	57.06%	63.28%	71.75%
All (Early Fusion)	58.76%	68.36%	70.06%
All (Combiner)	56.50%	63.28%	**72.88**%

Table 2.9 Agreement among three human annotators on text, audio, silent video, and full video modalities.

Modality	Agreement	Kappa
Text	30.76%	0.014
Audio	53.84%	0.040
Silent video	53.84%	0.040
Full video	53.84%	0.050

Table 2.10 Classification accuracy of three annotators (A1, A2, A3) and the developed systems on the real-deception dataset over four modalities.

	Text	Audio	Silent video	Full video
A1	69.23%	69.23%	69.23%	61.53%
A2	53.84%	61.53%	61.53%	76.92%
A3	69.23%	76.92%	76.92%	76.92%
Average	64.10%	69.22%	69.22%	**71.79**%
Fully-autom. sys.	71.75%	63.28%	61.58%	**72.88**%
Semi-autom. sys.	64.41%	63.28%	**80.79**%	**75.71**%

CHAPTER 3

DEEP NEURAL NETWORKS FOR SINGLE CHANNEL SOURCE SEPARATION

Performances of speech processing solutions, in general, are sensitive to the quality of the recordings such as amount of background noise or inclusion of different sound sources. Extracting fundamental frequency (pitch frequency) of the speech signal, which is used as a feature in our deception system, is also sensitive to the background sounds. In addition, lexical features are obtained from the transcriptions that are extracted using an automatic speech recognition system and automatic speech recognition systems are also known to be affected by artifacts that are present in the speech recording. In this regard, we believe, source separation of speech recordings would be a fundamental part of a fully automatic deception detection system.

In this chapter, a novel approach for single channel source separation (SCSS) using a deep neural network (DNN) architecture is introduced. Unlike previous studies in which DNN and other classifiers were used for classifying time-frequency bins to obtain hard masks for each source, we use the DNN to classify estimated source spectra to check for their validity during separation. In the training stage, the training data for the source signals are used to train a DNN. In the separation stage, the trained DNN is utilized to aid in estimation of each source in the mixed signal. Single channel source separation problem is formulated as an energy minimization problem where each source spectra estimate is encouraged to fit the trained DNN model and the mixed signal spectrum is encouraged to be written as a weighted sum of the estimated source spectra. The proposed approach works regardless of the energy scale differences between the source signals in the training and separation stages. Nonnegative matrix factorization (NMF) is used to initialize the DNN estimate for each source. The experimental results show that using DNN initialized by NMF for source separation improves the quality of the separated signal compared

with using NMF for source separation. [1]

3.1 Introduction

Single channel audio source separation is an important and challenging problem and has received considerable interest in the research community in recent years. Since there is limited information in the mixed signal, usually one needs to use training data for each source to model each source and to improve the quality of separation. In this work, we introduce a new method for improved source separation using nonlinear models of sources trained using a deep neural network.

3.1.1 Related work

Many approaches have been introduced so far to solve the single channel source separation problem. Most of those approaches strongly depend on training data for the source signals. The training data can be modeled using probabilistic models such as Gaussian mixture model (GMM) (Kristjansson, Attias & Hershey, 2004; Reddy & Raj, 2004,0), hidden Markov model (HMM) or factorial HMM (Deoras & Hasegawa-Johnson, 2004; Roweis, 2001; Virtanen, 2006). These models are learned from the training data and usually used in source separation under the assumption that the sources appear in the mixed signal with the same energy level as they appear in the training data. Fixing this limitation requires complicated computations as in (Hershey, Rennie, Olsen & Kristjansson, 2010; Ozerov, Févotte & Charbit, 2009; Radfar & Dansereau, 2007; Radfar, Wong, Chan & Dansereau, 2009; Radfar, Dansereau & Chan, 2010; Radfar, Wong, Dansereau & Chan, 2010). Another approach to model the training data is to train nonnegative dictionaries for the source signals (Grais & Erdogan, 2011b,1; Schmidt & Olsson, 2006). This approach is more flexible with no limitation related to the energy differences between the source signals in training and separation stages. The main problem in this approach is that any nonnegative linear combination of the trained dictionary vectors is a valid estimate for a source signal which may decrease the quality of separation. Modeling the training

[1] Work at this chapter is presented at IEEE International Conference on Acoustics, Speech and Signal Processing, 2014 (Grais, Sen & Erdogan, 2014)

data with both nonnegative dictionary and cluster models like GMM and HMM was introduced in (Grais & Erdoğan, 2012; Grais & Erdogan, 2012a,1; Grais & Erdoğan, 2013) to fix the limitation related to the energy scaling between the source signals and training more powerful models that can fit the data properly. Another type of approach which is called classification-based speech separation aims to find hard masks where each time-frequency bin is classified as belonging to either of the sources. For example in (Wang & Wang, 2012), various classifiers based on GMM, support vector machines, conditional random fields, and deep neural networks were used for classification.

3.1.2 Contributions

In this work, we model the training data for the source signals using a single joint deep neural network (DNN). The DNN is used as a spectral domain classifier which can classify its input spectra into each possible source type. Unlike classification-based speech separation where the classifiers are used to segment time-frequency bins into sources, we can obtain soft masks using our approach. Single channel source separation problem is formulated as an energy minimization problem where each source spectral estimate is encouraged to fit the trained DNN model and the mixed signal spectrum is encouraged to be written as a weighted sum of the estimated source spectra. Basically, we can think of the DNN as checking whether the estimated source signals are lying in their corresponding nonlinear manifolds which are represented by the trained joint DNN. Using a DNN for modeling the sources and handling the energy differences in training and testing is considered to be the main novelty of this chapter. Deep neural network (DNN) is a well known model for representing the detailed structures in complex real-world data (Hinton, Deng, Yu, Dahl, Mohamed, Jaitly, Senior, Vanhoucke, Nguyen, Sainath & others, 2012; Hinton, Osindero & Teh, 2006). Another novelty of this chapter is using nonnegative matrix factorization (Grais & Erdogan, 2011a) to find initial estimates for the sources rather than using random initialization.

3.1.3 Organization of the chapter

This chapter is organized as follows: In Section 3.2 a mathematical formulation for the SCSS problem is given. Section 3.3 briefly describes the NMF method for source separation. In Section 3.4, we introduce our new method. We present our experimental results in Section 3.5. We conclude the chapter in Section 3.6.

3.2 Problem formulation

In single channel source separation problems, the aim is to find estimates of source signals that are mixed on a single channel $y(t)$. For simplicity, in this work we assume the number of sources is two. This problem is usually solved in the short time Fourier transform (STFT) domain. Let $Y(t,f)$ be the STFT of $y(t)$, where t represents the frame index and f is the frequency-index. Due to the linearity of the STFT, we have:

$$Y(t,f) = S_1(t,f) + S_2(t,f), \tag{3.1}$$

where $S_1(t,f)$ and $S_2(t,f)$ are the unknown STFT of the first and second sources in the mixed signal. In this framework (Schmidt & Olsson, 2006; Virtanen, 2007), the phase angles of the STFT were usually ignored. Hence, we can approximate the magnitude spectrum of the measured signal as the sum of source signals' magnitude spectra as follows:

$$|Y(t,f)| \approx |S_1(t,f)| + |S_2(t,f)|\,. \tag{3.2}$$

We can write the magnitude spectrogram in matrix form as follows:

$$\mathbf{Y} \approx \mathbf{S}_1 + \mathbf{S}_2. \tag{3.3}$$

where $\mathbf{S}_1, \mathbf{S}_2$ are the unknown magnitude spectrograms of the source signals and need to be estimated using the observed mixed signal and the training data.

3.3 NMF for supervised source separation

In this section, we briefly describe the use of nonnegative matrix factorization (NMF) for supervised single channel source separation. We will relate our model to the NMF idea and we will use the source estimates obtained from using NMF as initilization for our method, so it is appropriate to introduce the use of NMF for source separation first.

To find a suitable initialization for the sources signals, we use nonnegative matrix factorization (NMF) as in (Grais & Erdogan, 2011a). NMF (Lee & Seung, 2001) factorizes any nonnegative matrix $\mathbf{V}$ into a basis matrix (dictionary) $\mathbf{B}$ and a gain matrix $\mathbf{G}$ as

$$\mathbf{V} \approx \mathbf{BG}. \tag{3.4}$$

The matrix $\mathbf{B}$ contains the basis vectors that are optimized to allow the data in $\mathbf{V}$ to be approximated as a linear combination of its constituent columns. The solution for $\mathbf{B}$ and $\mathbf{G}$ can be found by minimizing the following Itakura-Saito (IS) divergence cost function (Févotte, Bertin & Durrieu, 2009):

$$\min_{\mathbf{B},\mathbf{G}} D_{IS}\left(\mathbf{V}\,||\,\mathbf{BG}\right), \tag{3.5}$$

where

$$D_{IS}\left(\mathbf{V}\,||\,\mathbf{BG}\right) = \sum_{a,b}\left(\frac{\mathbf{V}_{a,b}}{(\mathbf{BG})_{a,b}} - \log\frac{\mathbf{V}_{a,b}}{(\mathbf{BG})_{a,b}} - 1\right).$$

This divergence cost function is a good measurement for the perceptual difference between different audio signals (Févotte et al., 2009). The IS-NMF solution for equation (3.5) can be computed by alternating multiplicative updates of $\mathbf{G}$ and $\mathbf{B}$ as follows:

$$\mathbf{G} \leftarrow \mathbf{G} \otimes \frac{\mathbf{B}^T\left(\frac{\mathbf{V}}{(\mathbf{BG})^2}\right)}{\mathbf{B}^T\left(\frac{\mathbf{1}}{\mathbf{BG}}\right)}, \tag{3.6}$$

$$\mathbf{B} \leftarrow \mathbf{B} \otimes \frac{\left(\frac{\mathbf{V}}{(\mathbf{BG})^2}\right)\mathbf{G}^T}{\left(\frac{\mathbf{1}}{\mathbf{BG}}\right)\mathbf{G}^T}, \tag{3.7}$$

where $\mathbf{1}$ is a matrix of ones with the same size of $\mathbf{V}$, the operation $\otimes$ is an element-wise multiplication, all divisions and $(.)^2$ are element-wise operations. The matrices $\mathbf{B}$ and $\mathbf{G}$ are usually initialized by positive random numbers and then updated iteratively using equations (3.6) and (3.7).

In the initialization stage, NMF is used to decompose the frames for each source i into a multiplication of a nonnegative dictionary $\mathbf{B}_i$ and a gain matrix $\mathbf{G}_i$ as follows:

$$\mathbf{S}_i^{train} \approx \mathbf{B}_i \mathbf{G}_i^{train}, \quad \forall i \in \{1,2\}, \tag{3.8}$$

where $\mathbf{S}_i^{train}$ is the nonnegative matrix that contains the spectrogram frames of the training data of source i. After observing the mixed signal, we calculate its spectrogram $\mathbf{Y}_{psd}$. NMF is used to decompose the mixed signal's spectrogram matrix $\mathbf{Y}_{psd}$ with the trained dictionaries as follows:

$$\mathbf{Y}_{psd} \approx [\mathbf{B}_1, \mathbf{B}_2]\,\mathbf{G} \quad \text{or} \quad \mathbf{Y}_{psd} \approx [\mathbf{B}_1 \quad \mathbf{B}_2] \begin{bmatrix} \mathbf{G}_1 \\ \mathbf{G}_2 \end{bmatrix}. \tag{3.9}$$

The only unknown here is the gains matrix $\mathbf{G}$ since the dictionaries are fixed. The update rule in equation (3.6) is used to find $\mathbf{G}$. After finding the value of $\mathbf{G}$, the initial estimate for each source magnitude spectrogram is computed as follows:

$$\hat{\mathbf{S}}_{init_1} = \frac{\mathbf{B}_1\mathbf{G}_1}{\mathbf{B}_1\mathbf{G}_1 + \mathbf{B}_2\mathbf{G}_2} \otimes \mathbf{Y}, \quad \hat{\mathbf{S}}_{init_2} = \frac{\mathbf{B}_2\mathbf{G}_2}{\mathbf{B}_1\mathbf{G}_1 + \mathbf{B}_2\mathbf{G}_2} \otimes \mathbf{Y}, \tag{3.10}$$

where $\otimes$ is an element-wise multiplication and the divisions are done element-wise.

The magnitude spectrograms of the initial estimates of the source signals are used to initialize the sources in the separation stage of the DNN approach.

3.4 Method

In NMF, the basic idea is to model each source with a dictionary, so that source signals appear in the nonnegative span of this dictionary. In the separation stage, the mixed signal is expressed as a nonnegative linear combination of the source dictionaries and separation is performed by taking the parts corresponding to each source in the decomposition.

The basic problem in NMF is that each source is modeled to lie in a nonnegative cone defined by all the nonnegative *linear* combinations of its dictionary entries. This assumption may be a limiting assumption usually since the variability within each source indicates that nonlinear models may be more appropriate. This limitation led us to consider nonlinear models for each source. It is not trivial to use nonlinear models or classifiers in source separation. Since deep neural networks were recently used with increased success in speech recognition and other object recognition tasks, they can be considered as superior models of highly variable real-world signals.

We first train a DNN to model each source in the training stage. We then use an energy minimization objective to estimate the sources and their gains during the separation stage. Each stage is explained below.

3.4.1 Training the DNN

We train a DNN that can classify sources present in the mixed signal. The input to the network is a frame of normalized magnitude spectrum, $\mathbf{x} \in \mathbb{R}^d$. The neural network architecture is illustrated in Figure 3.1. There are two outputs in the DNN, each corresponding to a source. The label of each training instance is a binary indicator function, namely if the instance is from source one, the first output label $f_1(\mathbf{x}) = 1$ and the second output label $f_2(\mathbf{x}) = 0$. Let n_k be the number of hidden nodes in layer k for $k = 0, \ldots, K$ where K is the number of layers. Note that $n_0 = d$ and $n_K = 2$. Let $\mathbf{W}_k \in \mathbb{R}^{n_k \times n_{k-1}}$ be the weights between layers $k-1$ and k, then the values of a hidden layer $\mathbf{h}_k \in \mathbb{R}^{n_k}$ are obtained as follows:

$$\mathbf{h}_k = g(\mathbf{W}_k \mathbf{h}_{k-1}), \tag{3.11}$$

where $g(x) = \frac{1}{1+\exp(-x)}$ is the elementwise sigmoid function. We skip the bias terms to avoid clutter in our notation. The input to the network is $h_0 = \mathbf{x} \in \mathbb{R}^d$ and the output is $f(\mathbf{x}) = \mathbf{h}_K \in \mathbb{R}^2$.

Training a deep network necessitates a good initialization of the parameters. It is shown that layer-by-layer pretraining using unsupervised methods for initialization of the parameters results in superior performance as compared to using random initial values. We used Restricted Boltzmann Machines (RBM) for initialization. After initialization, supervised backpropagation algorithm is applied to fine-tune the parameters. The learning criteria we use is least-squares minimization. We are able to get the partial derivatives with respect to the inputs, and this derivative is

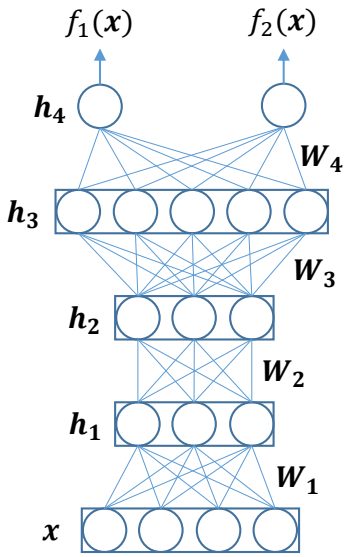


Figure 3.1 Illustration of the DNN architecture.

also used in the source separation part. Let $f(.) : \mathbb{R}^d \to \mathbb{R}^2$ be the DNN, then $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ are the scores that are proportional to the probabilities of source one and source two respectively for a given frame of normalized magnitude spectrum $\mathbf{x}$. We use these functions to measure how much the separated spectra carry the characteristics of each source as we elaborate more in the next section.

3.4.2 Source separation using DNN and energy minimization

In the separation stage, our algorithm works independently in each frame of the mixed audio signal. For each frame of the mixed signal spectrum, we calculate the normalized magnitude spectrum $\mathbf{y}$. We would like to express $\mathbf{y} = u\mathbf{x}_1 + v\mathbf{x}_2$ where u and v are the gains and $\mathbf{x}_1$ and $\mathbf{x}_2$ are normalized magnitude spectra of source one and two respectively.

We formulate the problem of finding the unknown parameters $\theta = (\mathbf{x}_1, \mathbf{x}_2, u, v)$ as an energy minimization problem. We have a few different criteria that the source estimates need to satisfy. First, they must fit well to the DNN trained in the training stage. Second, their linear combination must sum to the mixed spectrum $\mathbf{y}$ and third the source estimates must be nonnegative since they correspond to the magnitude spectra of each source.

The energy functions E_1 and E_2 below are least squares cost functions that quantify

the fitness of a source estimate $\mathbf{x}$ to each corresponding source model in the DNN.

$$E_1(\mathbf{x}) = (1 - f_1(\mathbf{x}))^2 + (f_2(\mathbf{x}))^2, \tag{3.12}$$

$$E_2(\mathbf{x}) = (f_1(\mathbf{x}))^2 + (1 - f_2(\mathbf{x}))^2. \tag{3.13}$$

Basically, we expect to have $E_1(\mathbf{x}) \approx 0$ when $\mathbf{x}$ comes from source one and vice versa. We also define the following energy function which quantifies the energy of error caused by the least squares difference between the mixed spectrum $\mathbf{y}$ and its estimate found by linear combination of the two source estimates $\mathbf{x}_1$ and $\mathbf{x}_2$:

$$E_{err}(\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}, u, v) = ||u\mathbf{x}_1 + v\mathbf{x}_2 - \mathbf{y}||^2. \tag{3.14}$$

Finally, we define an energy function that measures the negative energy of a variable, $R(\theta) = (\min(\theta, 0))^2$.

In order to estimate the unknowns in the model, we solve the following energy minimization problem.

$$(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \hat{u}, \hat{v}) = \underset{\{\mathbf{x}_1, \mathbf{x}_2, u, v\}}{\operatorname{argmin}} E(\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}, u, v), \tag{3.15}$$

where

$$\begin{aligned} E(\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}, u, v) = E_1(\mathbf{x}_1) + E_2(\mathbf{x}_2) + \lambda E_{err}(\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}, u, v) \\ + \beta \sum_i R(\theta_i) \end{aligned} \tag{3.16}$$

is the joint energy function which we seek to minimize. λ and β are regularization parameters which are chosen experimentally. Here $\theta = (\mathbf{x}_1, \mathbf{x}_2, u, v) = [\theta_1, \theta_2, \ldots, \theta_n]$ is a vector containing all the unknowns which must all be nonnegative. Note that, the nonnegativity can be given as an optimization constraint as well, however we obtained faster solution of the optimization problem if we used the negative energy function penalty instead. If some of the parameters are found to be negative after the solution of the optimization problem (which rarely happens), we set them to zero. We used the LBFGS algorithm for solving the unconstrained optimization problem.

We need to calculate the gradient of the DNN outputs with respect to the input $\mathbf{x}$ to be able to solve the optimization problem. The gradient of the input $\mathbf{x}$ with

respect to $f_i(\mathbf{x})$ is given as $\frac{\partial f_i(\mathbf{x})}{\partial \mathbf{x}} = \mathbf{q}_{1,i}$ for $i = 1, 2$, where,

$$\mathbf{q}_{k,i} = \mathbf{W}_k^T(\mathbf{q}_{k+1,i} \otimes \mathbf{h}_k \otimes (1 - \mathbf{h}_k)), \tag{3.17}$$

and $\mathbf{q}_{K,i} = f_i(\mathbf{x})(1 - f_i(\mathbf{x}))\mathbf{w}_{K,i}^T$, where $\mathbf{w}_{K,i} \in \mathbb{R}^{n_{K-1}}$ contains the weights between i^{th} node of the output layer and the nodes at the previous layer, in other words the ith row of $\mathbf{W}_K$.

The flowchart of the energy minimization setup is shown in Figure 3.2. For illustration, we show the single DNN in two separate blocks in the flowchart. The fitness energies are measured using the DNN and the error energy is found from the summation requirement.

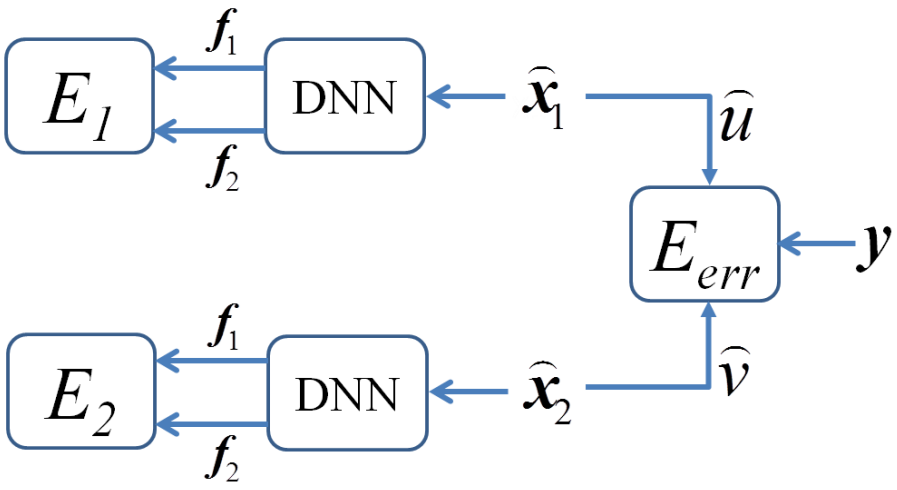


Figure 3.2 Flowchart of the energy minimization setup. For illustration, we show the single DNN in two separate blocks in the flowchart.

Note that, since there are many parameters to be estimated and the problem is clearly non-convex, the initialization of the parameters is very important. We initialize the estimates $\hat{\mathbf{x}}_1$ and $\hat{\mathbf{x}}_2$ from the NMF result after normalizing by their ℓ_2-norms. $\hat{u}$ is initialized by the ℓ_2-norm of the initial NMF source estimate $\hat{\mathbf{s}}_1$ divided by the ℓ_2-norm of the mixed signal $\mathbf{y}$. $\hat{v}$ is initialized in a similar manner.

After we obtain $(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \hat{u}, \hat{v})$ as the result of the energy minimization problem, we use them as spectral estimates in a Wiener filter to reconstruct improved estimates

of each source spectra, *e.g.* for source one we obtain the final estimate as follows:

$$\hat{\mathbf{s}}_1 = \frac{(\hat{u}\hat{\mathbf{x}}_1)^2}{(\hat{u}\hat{\mathbf{x}}_1)^2 + (\hat{v}\hat{\mathbf{x}}_2)^2} \otimes \mathbf{y}. \tag{3.18}$$

3.5 Experiments and Discussion

We applied the proposed algorithm to separate speech and music signals from their mixture. We simulated our algorithm on a collection of speech and piano data at 16kHz sampling rate. For speech data, we used the training and testing male speech data from the TIMIT database. For music data, we downloaded piano music data from piano society web site (Society, 2020). We used 39 pieces with approximate 185 minutes total duration from different composers but from a single artist for training and left out one piece for testing. The magnitude spectrograms for the speech and music data were calculated using the STFT: A Hamming window with 480 points length and 60% overlap was used and the FFT was taken at 512 points, the first 257 FFT points only were used since the conjugate of the remaining 255 points are involved in the first points.

The mixed data was formed by adding random portions of the test music file to 20 speech files from the test data of the TIMIT database at different speech to music ratio. The audio power levels of each file were found using the "speech voltmeter" program from the G.191 ITU-T STL software suite (software, 2014).

For the initialization of the source signals using nonnegative matrix factorization, we used a dictionary size of 128 for each source. For training the NMF dictionaries, we used 50 minutes of data for music and 30 minutes of the training data for speech. For training the DNN, we used a total 50 minute subset of music and speech training data for computational reasons.

For the DNN, the number of nodes in each hidden layer were 100-50-200 with three hidden layers. Sigmoid nonlinearity was used at each node including the output nodes. DNN was initialized with RBM training using contrastive divergence. We used 150 epochs for training each layer's RBM. We used 500 epochs for backpropagation training. The first five epochs were used to optimize the output layer keeping the lower layer weights untouched.

In the energy minimization problem, the values for the regularization parameters were $\lambda = 5$ and $\beta = 3$. We used Mark Schmidt's minFunc matlab LBFGS solver for solving the optimization problem (Schmidt, 2014).

Performance measurements of the separation algorithm were done using the signal to distortion ratio (SDR) and the signal to interference ratio (SIR) (Vincent, Gribonval & Févotte, 2006). The average SDR and SIR over the 20 test utterances are reported. The source to distortion ratio (SDR) is defined as the ratio of the target energy to all errors in the reconstructed signal. The target signal is defined as the projection of the predicted signal onto the original speech signal. Signal to interference ratio (SIR) is defined as the ratio of the target energy to the interference error due to the music signal only. The higher SDR and SIR we measure the better performance we achieve. We also use the output SNR as additional performance criteria.

The results are presented in Tables 3.1 and 3.2. We experimented with multi-frame DNN where the inputs to the DNN were taken from L neighbor spectral frames for both training and testing instead of using a single spectral frame similar to (Grais & Erdogan, 2011b). We can see that using the DNN and the energy minimization idea, we can improve the source separation performance for all input speech-to-music ratio (SMR) values from -5 to 5 dB. In all cases, DNN is better than regular NMF and the improvement in SDR and SNR is usually around 1-1.5 dB. However, the improvement in SIR can be as high as 3 dB which indicates the fact that the introduced method can decrease remaining music portions in the reconstructed speech signal. We performed experiments with $L = 3$ neighboring frames which improved the results as compared to using a single frame input to the DNN. For $L = 3$, we used 500 nodes in the third layer of the DNN instead of 200. We conjecture that better results can be obtained if higher number of neighboring frames are used.

Table 3.1 SDR, SIR and SNR in dB for the estimated speech signal.

SMR	NMF			DNN					
				$L=1$			$L=3$		
dB	SDR	SIR	SNR	SDR	SIR	SNR	SDR	SIR	SNR
-5	1.79	5.01	3.15	2.81	7.03	3.96	**3.09**	**7.40**	**4.28**
0	4.51	8.41	5.52	5.46	9.92	6.24	**5.73**	**10.16**	**6.52**
5	7.99	12.36	8.62	8.74	**13.39**	9.24	**8.96**	13.33	**9.45**

Table 3.2 SDR, SIR and SNR in dB for the estimated music signal.

SMR	NMF			DNN					
				$L=1$			$L=3$		
dB	SDR	SIR	SNR	SDR	SIR	SNR	SDR	SIR	SNR
-5	5.52	15.75	6.30	6.31	**18.48**	7.11	**6.67**	18.30	**7.43**
0	3.51	12.65	4.88	4.23	**16.03**	5.60	**4.45**	15.90	**5.88**
5	0.93	9.03	3.35	1.79	12.94	3.96	**1.97**	**13.09**	**4.17**

3.6 Conclusion

In this work, we introduced a novel approach for single channel source separation (SCSS) using deep neural networks (DNN). The DNN was used in this work as a helper to model each source signal. The training data for the source signals were used to train a DNN. The trained DNN was used in an energy minimization framework to separate the mixed signals while also estimating the scale for each source in the mixed signal. Many adjustments for the model parameters can be done to improve the proposed SCSS using the introduced approach. Different types of DNN such as deep autoencoders and deep recurrent neural networks which can handle the temporal structure of the source signals can be tested on the SCSS problem. We believe this idea is a novel idea and many improvements will be possible in the near future to improve its performance.

CHAPTER 4

LEARNING WORD REPRESENTATIONS FOR TURKISH

In Chapter-2, we showed that word embeddings that are pretrained on large corpora in an unsupervised manner resulted in the best performance among the lexical feature sets. Implementing a deception system for Turkish would only require training word embeddings for Turkish, as the other two modalities are language independent for the most part. However, training and employing word embeddings in Turkish is not straightforward because of the morphologically rich structure of Turkish. In this chapter, we investigate the usage of word embeddings in Turkish and measure their performances on general semantic and syntactic tasks. We believe, this work would be useful for the development of an automatic deception detection system in Turkish.

High-quality distributed word representations have been very successful in recent years at improving performance across a variety of NLP tasks. These word representations are the mappings of each word in the vocabulary to a real vector in the Euclidean space. Besides high performance on specific tasks, learned word representations have been shown to perform well on establishing linear relationships among words. The recently introduced Skip-gram model improved performance on unsupervised learning of word embeddings that contains rich syntactic and semantic word relations. Word embeddings that have been used frequently on English language, is not applied to Turkish prior to this work. In this work, we apply the Skip-gram model to a large Turkish text corpus and measured the performance of them quantitatively with the "question" sets that we generated. The learned word embeddings and the question sets are publicly available at our website. [1]

[1] Work at this chapter is presented at 22nd Signal Processing and Communications Applications Conference (SIU) (Sen & Erdogan, 2014)

4.1 Introduction

Successes of Deep Neural Network (DNN) based solutions on Natural Language Processing (NLP) problems are reported with a vast amount of publications in recent years. However, DNN based methods that solve NLP problems, such as Named Entity Recognition, Part of Speech Tagging, Semantic Role Labeling, are mostly employed on English datasets. A significant process of such methods is unsupervised learning of word embeddings that map words to low dimensional continuous real vectors from large text corpora. In such unsupervised models, each word (w) is represented by a d-dimensional real vector ($v_w \in \Re^d$). Word embeddings encode semantic and syntactic information into low dimensional vectors.

Word embeddings are employed in Deep Neural Network based models; such as Recurrent Neural Networks (Bengio, Ducharme, Vincent & Janvin, 2003), hierarchical models (Mnih & Hinton, 2008; Morin & Bengio, 2005) or Convolutional Neural Networks (Mikolov, Karafiát, Burget, Cernocký & Khudanpur, 2010; Mikolov, Kombrink, Burget, Cernocký & Khudanpur, 2011); as well as more customary feature-based methods (Koo, Carreras & Collins, 2008; Ratinov & Roth, 2009). Collobert et al. used word embeddings as inputs to a multi-objective NLP model and reported improved performance over single-objective models (Collobert & Weston, 2008). In another work, learned word representations are observed to hold linear semantic relationships (Mikolov, tau Yih & Zweig, 2013). For example, the difference between the vectors for the words *King* and *Queen* is reported to be very close to that of the words *Man* and *Woman*. Based on this observation, Mikolov et al. prepared questions for measuring the performance of word embeddings quantitatively, not for a particular NLP problem but in the sense of linear semantic and syntactic information capacity of the words.

Word embeddings that are frequently and successfully used in English language had not been used in Turkish prior to this work to the best of our knowledge. This work learns word embeddings from a large Turkish corpus with the aim of investigating the usefulness of word embeddings in Turkish and draw attention from Turkish NLP community. The quality of the word embeddings that are learned using the skip-gram model (Mikolov, Chen, Corrado & Dean, 2013a; Mikolov, Sutskever, Chen, Corrado & Dean, 2013a) is measured quantitavely using constructed Turkish question-sets. Learned word representations and constructed question sets are publicaly shared online. [2]

[2] http://myweb.sabanciuniv.edu/umutsen/research/

4.2 Skip-Gram Model

In skip-gram model, word in a sentence is given to the log-linear classifier as input and the nearby words are estimated (Mikolov et al., 2013a,1). Input word is mapped to a vector using a look-up table and given to the classifier as input. Parameters of the look-up table constitutes the word embeddings. Let $w_1, w_2, \ldots, w_T$ be a sequence of words, then the skip-gram model maximizes the following objective function:

$$\Phi = \frac{1}{T} \sum_{t=1}^{T} \sum_{-c \leq j \leq c, j \neq 0} \log p(w_{t+j} | w_t) \tag{4.1}$$

where, $c/2$ is the number of context words that are to be estimated and it is called context size. Increasing c would lead to higher quality word embeddings but training time is increased too. For the conditional $p(w_{t+j}|w_t)$, softmax formulation is used:

$$p(w_O | w_I) = \frac{\exp(\mathbf{y}_{w_O}^T \mathbf{v}_{w_I})}{\sum_{w=1}^{W} \exp(\mathbf{y}_w^T \mathbf{v}_{w_I})} \tag{4.2}$$

where, $\mathbf{v}_w$ and $\mathbf{y}_w$, are the input and output word embedding vectors of the word w and W is the vocabulary size. After training, $\mathbf{v}_w$ is used as the word representation vector. Calculation of the denominator in the above formulation is computationally expensive and two of the solutions of this problem are Hierarchical Softmax and Negative Sampling.

4.2.1 Hierarchical Softmax

This method is a computationally efficient approximation to the softmax function and employs binary trees (Morin & Bengio, 2005). Leaf nodes in the tree represent the words in vocabulary and there are vector representations for the internal nodes too. For each word, there is a certain path from the root. The conditional probability defined in (4.1) of a word is found by taking inner products of the input vector with every node along the path from the root to the leaf, taking the sigmoid function and multiplying all resulting scalar values. This results in approximately $\log_2(W)$

evaluations instead of W evaluations in regular softmax calculations. In addition, each word has a single vector representation in Hierarchical Softmax method unlike in the original skip-gram formulation in which words have 2 vector representations. Structure of the tree effects the performance of the model and Binary Huffman Tree is shown to work well for learning word embeddings (Mikolov et al., 2011,1; Mnih & Hinton, 2008) .

4.2.2 Negative Sampling

Negative Sampling is a simplified form of Noise Contrastive Estimation and uses the following conditional probability model:

$$\log p(w_o|w_I) = \log \sigma(\mathbf{y}_{w_o}^T \mathbf{v}_{w_I}) + \sum_{i=1}^{k} \log \sigma(-\mathbf{y}_{w_i}^T \mathbf{v}_{w_I}) \tag{4.3}$$

where, w_i is a randomly selected word from vocabulary, and $\sigma(x) = 1/(1+\exp(-x))$ is the sigmoid function. This method samples k words from vocabulary and use them as negative samples for a given a pair of input and output words. Second term in the right side of the above equation uses these negative samples.

In general, Negative Sampling works better than Hierarchical Softmax in terms of accuracy and computational convergence speed and our experiments showed consistent results as shown in Section-4.3.

4.2.3 Subsampling Frequent Words

A subset of the words has very large frequencies in the dataset ("*ve*":10 million, "*bir*":9 million). Such high-frequency words may not have as much mutual information as less frequent words with a given sample word. In addition, vector representations of high-frequent words do not change significantly after a certain amount of training. In order to balance between rare and very frequent words, we used the method at Mikolov et al. (Mikolov et al., 2013a): each word is removed from the

training data with the following formula:

$$p(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}} \tag{4.4}$$

where, $f(w_i)$ is the occurrence frequency of the word and t is a threshold with a chosen value of 0.001.

4.3 Experiments

We used two different corpora for training word representations. First one is the Turkish Wikipedia dump (wikipedia, 2013), second one is a Turkish online news corpus, which we call Bogazici Corpus throughout the chapter (Sak, Güngör & Saraçlar, 2008). Wikipedia corpus contains 52 million words after removing irregular content such as tables and figures. Bogazici corpus contains 270 million words.

4.3.1 Preprocessing

First, corpora are cleaned from punctuation and non-text elements. Separating suffixes from the roots affects the performance of the quality of word embeddings because of the high morphological structure of the Turkish language. Hence, words that occur in the corpora less than 1000 are separated into roots and suffixes using the Zemberek toolkit (Akin & , 2007) and the rest of the high frequency words are left as they appear in the dataset. Words that are not recognized with the morphological analyzer of the Zemberek toolkit are attempted to be deasscified using again the Zemberek toolkit. These words, then, again applied to morphological analyzer and the words that are failed to be recognized are left as they appear in the dataset. In the case that the analyzer resulted in a multiple choice for separation, the option with the longest root is chosen. After finding the root, rest of the word part is employed as a single suffix, ignoring the possible transformations of the suffixes. For example the word *geldiklerinde* is transformed into two tokens as *gelmek* + *_diklerinde*.

After the morphological transformation, vocabulary is constructed and words that appear less than 6 in the dataset are discarded. In the final dataset, there are 530 million words and the vocabulary size is approximately 380.000 of which approximately 25.000 are suffixes.

4.3.2 Quantitative Evaluation

We applied two different testing schemas for evaluating the quality of the learned word representations. First one is about finding analogical relations between word pairs and this method is introduced in the work that introduced the skip-gram model and applied for the English language (Mikolov et al., 2013a,1). Each test question is composed of 4 words and has the form "If $A \rightarrow B$, then $C \rightarrow$?. For finding the word in the place of the question mark, linear relationships of the word embeddings are used:

$$\mathbf{v}_{\hat{D}} = \mathbf{v}_B - \mathbf{v}_A + \mathbf{v}_C \tag{4.5}$$

After finding the vector of $\mathbf{v}_{\hat{D}}$, the word that has the closest embedding to this vector in terms of cosine similarity (excluding A, B, and C) is chosen to be the answer:

$$\hat{D} = \arg\max_{w} \frac{\mathbf{v}_w^T \mathbf{v}_{\hat{D}}}{\|\mathbf{v}_w\| \left\|\mathbf{v}_{\hat{D}}\right\|} \tag{4.6}$$

We prepared question sets for the analogical reasoning task in Turkish for evaluating semantic and syntactic performances separately. Definitions for these question sets and some examples are given at Table-4.1 and Table-4.2. We obtained a total of 26588 questions in these two categories.

Second testing category constitutes a semantic task: given a group of words, finding the single word that does not belong the the group. We find the incompatible word by first finding the mean embedding vector of the group and choosing the word with the highest distance to the mean vector in the Euclidean space. We have chosen the incompatible words such that they are semantically relatively close to the group with

Table 4.1 Semantic Analogy Question Sets

Set Name	# of Quest.	Sample
Family Relations	132	(kız → oğul) ⇔ (gelin → damat)
Capitals	2970	(Tokyo → Japonya) ⇔ (Brüksel → Belçika)
Synonyms	3422	(sözcük → kelime) ⇔ (ırmak → nehir)
Districts	6466	(Konak → İzmir) ⇔ (Beyoğlu → İstanbul)
Currencies	156	(ABD → dolar) ⇔ (Hindistan → rupi)
Antonyms	2756	(barış → savaş) ⇔ (büyük → küçük)

Table 4.2 Syntactic Analogical Question Sets

Set Name	# of Quest.	Sample
Plurals	4830	(olay → olaylar) ⇔ (işlem → işlemler)
Negatives	756	(sever → sevmez) ⇔ (döner → dönmez)
Past Time	3540	(bulmak → buldu) ⇔ (istemek → istedi)
Present Time	1560	(etkilemek → etkiler) ⇔ (yaşamak → yaşar)

Table 4.3 Group Question Sets

Countries	Units	Trees	Animals	Provinces
1029	476	31	198	438
Asya	ağırlık	meyve ağ.	kuşlar	Marmara
Afrika	uzunluk	diğer ağ.	balıklar	Ege
Doğu Avrupa	sıcaklık		sürüngenler	Akdeniz
Kuzey Avrupa	alan		memeliler	Karadeniz
Güney Avrupa	basınç			Doğu Anadolu
Batı Avrupa	hacim			Güney Doğu An.
Güney Amerika	zaman			İç Anadolu
Kuzey Amerika	bilg. hafızası			
	para			

the intent of making the task more difficult. For example, countries are separated with respect to the continents they are in and the country that is not in the same continent with the rest of the group is chosen to be the incompatible word (e.g. *England*, *France*, *The Netherlands*, *Ireland*, *Switzerland*, *Algeria*). *Group Question* sets are summarized at Table-4.3. There are 2172 group testing questions in total.

Table 4.4 Accuracies - Hierarchical Softmax and Negative Sampling

	Hierarchical Softmax				Negative Sampling			
	Group	Top-1	Top-3	Top-10	Group	Top-1	Top-3	Top-10
Semantic	-	23.02	35.77	49.33	-	29.69	43.18	56.48
Syntactic	-	28.67	43.58	58.75	-	42.25	58.97	72.78
General	58.83	25.29	38.91	53.11	57.50	34.74	49.52	63.03
Time	642 min.				459 min.			

4.3.3 Results

4.3.3.1 Method Comparisons

Firstly, We compare the Negative Sampling with the Hierarchical Softmax method using the generated questions sets. We used 5 as the number of negative samples for each example (Eq-4.3). The context size is chosen to be 5 for each method ($c = 5$ at Eq-4.1). Word embedding dimensions are chosen to be 200. Results are given at Table-4.4. Top-n results are for the analogical reasoning task and the data point is said to be correctly classified if the true word is in the top-n of the estimated words when sorted according to the similarity.

We see that Negative Sampling works better than Hierarchical Softmax both in terms of runtime and accuracy. Another result is that semantic accuracies are lower than syntactic accuracies in general. We attribute this result to that syntactic questions are mostly composed of verbs and the number of verbs in the dataset is higher than the other types of words. But another reason might be that semantic questions are more difficult than the syntactic questions in general. All the experiments are done with the Negative Sampling method from now on.

4.3.3.2 Removing Suffixes

We expect that removing suffixes from the datasets would improve the performances since the questions do not contain any suffixes. We compare the results using datasets with and without suffixes in Table-4.5.

Table 4.5 Accuracies using Datasets with and without Suffixes

	Without Suffixes				With Suffixes			
	Group	Top-1	Top-3	Top-10	Group	Top-1	Top-3	Top-10
Semantic	-	35.40	48.42	60.80	-	29.69	43.18	56.48
Syntactic	-	43.17	60.21	74.68	-	42.25	58.97	72.78
General	61.00	38.52	53.16	66.38	57.50	34.74	49.52	63.03
Time	453 min.				459 min.			

Table 4.6 Runtimes vs. Dimension Sizes

Vector Dimension	100	200	300	400	500	600	700
Time (min.)	299	453	539	735	899	994	1171

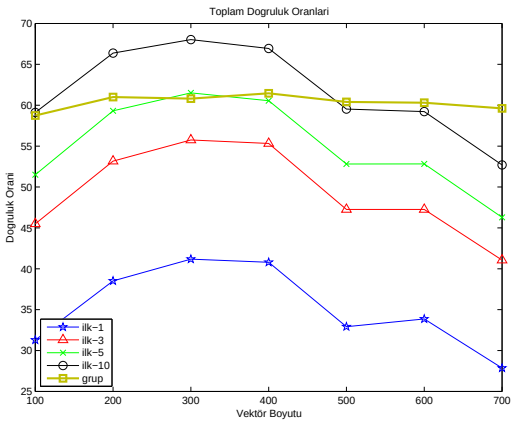
4.3.3.3 Effect of Vector Dimension

In this section, we analyzed the effect of word embeddings' dimension on semantic and syntactic accuracies. We used Negative Sampling method and the same parameters with the previous experiments. Results are depicted at Figure-4.1.

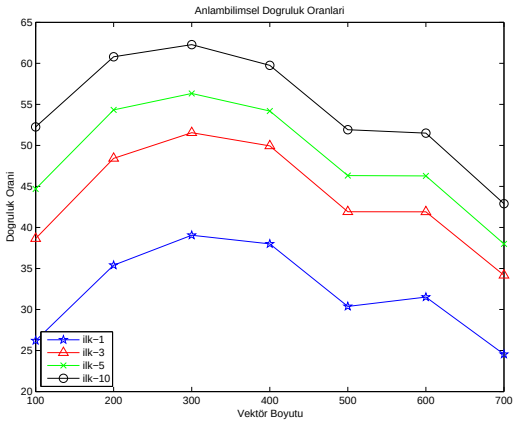
We observe consistent changes in accuracies when we change the vector dimensions. The dimension size is observed to have more effect on semantic accuracy than the syntactic accuracy. Runtimes of the learning for different vector sizes are given at Table-4.6 .

4.4 Conclusion and Future Work

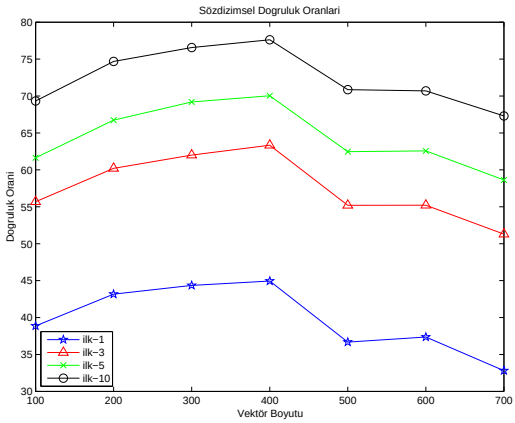
In this chapter, we trained word embeddings using a Turkish corpus. We created question sets for measuring the semantic and syntactic performances of word embeddings in a linear fashion. We believe this work had a good impact in the Turkish NLP community in terms of usage of unsupervised training of word embeddings, which are frequently and successfully used in English, for Turkish. We believe, unsupervised models that take into account the rich morphological structure of the Turkish language would have a high impact on various NLP problems.



(a) Total



(b) Semantic



(c) Syntactic

Figure 4.1 Change in accuracies (y-axis) with respect to vector dimensions (x-axis) for top-1, top-3, top-5, top-10 scorings.

CHAPTER 5

DOCUMENT CLASSIFICATION OF SUDER TURKISH NEWS CORPORA

Word embeddings are successfully employed in various Natural Language Processing tasks, but training them requires large amount of text, which is scarce for Turkish. In this work, we collected large amounts of articles from two news websites and tags within web pages are used as labels. Obtained corpora are tested with various document classification models. Embedding based models performed better than models with the traditional TF-IDF features. A neural network that simultaneously learns the word embeddings and document classification performed the best. [1]

5.1 Introduction

Document classification is the problem of automatically separating texts into categories. Some applications are subject classification, spam filtering and sentiment classification. In this chapter, we focus on subject classification.

Traditional approaches extract features from text using the occurrence statistics of words and uses Machine Learning models for classifying these features. Term-frequency Inverse-document-frequency (TF-IDF) features and its variants are the most used feature family that have successful results for various applications. A survey for different alterations in feature extraction and ML models is given at Jindal et al. (Jindal, Malhotra & Jain, 2015)

[1] Work at this chapter is presented at 26th Signal Processing and Communications Applications Conference (SIU) (Sen & Yanıkoğlu, 2018)

As introduced at Chapter-4, word embeddings are representations of words with low dimensional real vectors and has vast usage on various problems with success. These vectors are given to the Neural Networks (NN) as input and can be updated through Back Propagation (BP) algorithm. Goldberg surveyed the usage of NNs on NLP problems (Goldberg, 2016).

Mikolov et al. showed that unsupervised learning of word embeddings with large corpora contains semantic and syntactic information about words (Mikolov, Chen, Corrado & Dean, 2013b). Word embeddings that are learned using unsupervised models from large unlabeled corpora can be used for initialization of models that are trained with a small labeled corpus for a specific problem.

There are numerous works in the literature for Turkish text classification. Kilic et al. introduced two variants of TF-IDF and showed their success on Turkish datasets (Kilic, Ates, Karakaya & Sahin, 2015). Ay et al. used a genetic algorithm and introduced a new quality weighting method (Ay, Doğan, Alver & Kaya, 2016). Sahin showed that SVM classification of document representations that are obtained by averaging embeddings of words inside the documents overperformed the traditional TF-IDF based classification (Sahin, 2017). In this chapter, we showed that using NN as the classifier and updating the embeddings during training of the NN further improves the performance.

Available Turkish corpora for document classification have been increasing at recent years. Sahin et al. labeled Turkish Wikipedia articles in an automated fashion and obtained a corpus with approximately 10 million words (Sahin, Tirkaz, Yildiz, Eren & Sonmez, 2017). Tufekci et al. collected news articles from 5 different online news portals and obtained a corpus with 750 documents of 5 categories (Tüfekci, Uzun & Sevinç, 2012). They analyzed effects of different morphological processing methods on classification performance and they showed that removing words that are not nouns from the corpus allowed them to reduce the vector dimension substantially without sacrificing accuracies. Kilinc et al. collected news articles from 6 different online news portals and obtained a corpus of 3600 documents (Kılınç, Özçift, Bozyigit, Yıldırım, Yücalar & Borandag, 2017).

Successes of word embeddings are highly dependent on the corpus sizes. State-of-the-art accuracies are mostly obtained with word embeddings that are trained using very large corpora. Despite the increasing number of available Turkish text corpora, there is still a lack of very large Turkish text corpora. In this work, we obtained two large corpora from online news portals. We conveyed experiments for measuring performances of supervised text categorization models such as TF-IDF with Support Vector Machines (SVM), word embeddings with Neural Networks (NN) and Latent

Dirichlet Allocation which is an unsupervised topic modelling method.

5.2 Corpora

We downloaded news articles, opinion columns, art galleries and video sharing web pages from two online news portals, namely Sabah[2] and Cumhuriyet[3], and text are extracted from these web pages along with category, date, title information.

From Sabah, approximately 426.000 web pages obtained between dates 2010-January and 2017-July and pages with number of words less than 10 are discarded which resulted in 420513 pages in total. There are 4 different categories and descriptive statistics are given at Table-5.1. Titles are not included in these statistics as well as in the experiments.

From the Cumhuriyet portal, approximately 463.000 pages are obtained with dates up to 2017-September. Pages that are published before the year 2014 do not contain category information and they are discatded. The remaining number of pages with category information is approximately 273.000. After discarding the text with the number of words less than 10, 268.784 documents remained that belong to 14 different categories. Descriptions of the Cumhuriyet corpus is given at Table-5.2.

Table 5.1 Statistics for the Sabah Corpus

Category	**Num. of Documents**			**Num. of Words**	
	Total	**Train**	**Test**	**Total**	**Average**
gündem	143,842	117,019	26,823	35,749,880	248.54
yaşam	123,086	108,202	14,884	22,878,732	180.86
ekonomi	85,485	75,512	9,973	22,261,600	247.38
yazarlar	68,100	60,683	7,417	16,335,364	239.87
Total	420,513	361,416	59,097	95,494,110	227.09

We used documents published before September, 1, 2016 as the training set and rest of the documents as the test set. Suffixes that are added to the words with apostrophe are included in the vocabulary, single letter words and numbers are excluded from the vocabulary [4].

[2] www.sabah.com.tr

[3] www.cumhuriyet.com.tr

[4] Corpora are publicly available at: `https://github.com/suverim/suder`

Table 5.2 Statistics for the Cumhuriyet Corpus

Category	Num. of Documents			Num. of Words	
	Total	Train	Test	Total	Average
türkiye	84,741	56,140	28,524	22,829,220	269.39
yazarlar	33,835	29,694	4,141	16,663,717	492.49
video	33,409	23,686	9,723	2,007,691	60.09
spor	31,396	24,627	6,730	7,240,974	230.63
dünya	21,005	14,684	6,152	4,416,708	210.26
siyaset	15,969	11,274	4,686	6,409,811	401.39
foto	14,302	9,729	110	248,871	17.40
ekonomi	8,187	5,811	2,356	2,520,473	307.86
teknoloji	7,913	5,089	2,810	1,734,268	219.16
kültür-sanat	6,506	4,680	1,806	2,664,020	409.47
yaşam	4,833	3,931	886	918,754	190.10
sağlık	2,573	2,047	514	863,208	335.48
eğitim	2,380	1,544	805	744,396	312.77
çevre	1,735	1,081	607	477,811	275.39
Total	268,784	194,017	69,850	69,739,922	259.46

5.3 Methods

5.3.1 TF-IDF ve Support Vector Machines

Term Frequency-Inverse Document Frequency (TF-IDF) features represent each document with a fixed and large dimensional vector. Each dimension corresponds to a word in the vocabulary and values are proportional to the occurrence frequency of the corresponding word in the document. Let c_{dt} be the number of occurrences of the term-t in document-d and let N_d be the number of words in document-d. Term frequency (TF) is calculated as follows: $tf(d,t) = c_{dt}/N_d$. To eliminate the disturbing effect of very frequent words that do not contain information about the categories, such as stop words like "a" or "the", TF is multiplied by the Inverse Document Frequency:

$$tdf(t) = \log\left(\frac{1+D}{1+m_t}\right) \tag{5.1}$$

where, D is the number of documents, and m_t is the number of documents that

contain the term t. Then, TF-IDF is calculated as follows: $tftdf(d,t) = tf(d,t) \times tdf(t)$.

We used single words as terms in this work as opposed to including word phrases. We experimented with different values for the number of TF-IDF features ranging from 1000 to 50000. Term vectors are normalized using l_1 normalization. Obtained features are modeled using Support Vector Machines (SVM). We used the linear SVM learned with primal formulation since learning time of linear SVMs are much smaller than non-linear SVMs but the accuracy values are close to the non-linear SVMs when the number of examples is much larger than the number of features (Keerthi & DeCoste, 2005). "One-vs-all" method is used as the multi-class modification of the original binary SVM formulation (Rifkin & Klautau, 2004).

5.3.2 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is an unsupervised probabilistic model for the topic modelling problem (Blei, Ng & Jordan, 2003). In topic modelling, each document receives a topic distribution instead of a hard classification into one of the categories. In LDA, each word in a document is sampled from a single topic that is sampled from the topic distribution of the document and each topic has a distribution over the vocabulary. Number of topics is determined apriori.

In this work, we tried different number of topics (K) through experiments. Online LDA training method, that uses a Variational Bayes method for inference, is used since the corpus is large and Online LDA works faster than the regular LDA (Hoffman, Bach & Blei, 2010)[5]. We used the purity score for measuring the classification performance of the LDA topic model. After training the model, each topic is mapped to a specific category. For finding this mapping, topic distribution of each document is calculated (γ_{dk}: probability that document-d is a sample from topic-k); and each topic is assigned to the category with the maximum empirical expected probability:

$$m_k = \arg\max_c \frac{1}{|\mathcal{D}_c|} \sum_{d:d\in\mathcal{D}_c} \gamma_{dk} \tag{5.2}$$

[5]LDA code obtained from: `github.com/wellecks/online_lda_python`

where, $\mathcal{D}_c$ is the document set of class-c; and m_k represents the class of topic-k.

5.3.3 Word Embeddings and Support Vector Machines

Word embeddings that are learned from large corpora in an unsupervised setting are proven to contain semantic and syntactic information about the words (Mikolov et al., 2013b; Sen & Erdogan, 2014). In this work, we used the skip-gram model that is defined at Chapter-4 (Mikolov et al., 2013b; Mikolov, Sutskever, Chen, Corrado & Dean, 2013b). We used the Negative Sampling objective function.

We used the average of the word embeddings of the words in the document to obtain the document representation vector. Then document representations are fed into SVM model with the labels of the dataset. This method was shown to be successful for Turkish in the work of Sahin et al. (Sahin, 2017).

5.3.4 Word Embeddings and Neural Networks

In this method, we used a Neural Network (NN) for the modelling with the word embeddings. Let $\mathbf{w}_t \in \mathbb{R}^d$ be the word embedding of a word t and let $f : \mathbb{R}^d \rightarrow \mathbb{R}^C$ be a NN, where C is the number of classes. Then the classification of a document $S_d = \left\{t_1, \ldots, t_{N_d}\right\}$ is formulated as follows:

$$y_c(d) = f\left(\frac{1}{|S_d|}\sum_{t \in S_d} \mathbf{w}_t\right) \tag{5.3}$$

where, $y_c(d)$ is the score of class-c for the document-d. Sums of least-squares function is used as the objective:

$$\Phi = \frac{1}{CD}\sum_{d=1}^{D}\sum_{c=1}^{C}(y_c(d) - \delta_{dc})^2 \tag{5.4}$$

where, δ_{dc} is 1 if document-d belongs to class-c, 0 otherwise; and D is the total

number of documents in the training data. Unlike previous work that uses this setting such as (Sahin, 2017), we let the word embeddings to be updated during training with the Back-Propagation algorithm. This helps the word embeddings to be more discriminative features for separating document labels and results in better accuracies. It should be noted that, word embeddings that are learned in an unsupervised manner using skip-gram model are used as the initialization of the word-embedding layer of the NN.

5.4 Experiments

Some preprocessing steps that we applied are; lowercasing, separating suffixes that are added appended to words using apostrophe and removing them, discarding single-letter words, and discarding numbers. Stop words that are collected from various online resources[6] [7] [8] are also discarded. Number of stop-words that are listed is 553.

Zemberek toolkit is used for morphological processing (Akın & Akın, 2007). After morphological analysis, option with the longest root is chosen and all suffixes are discarded. Choosing longest root is shown to obtain better results in the literature before (Cataltepe, Turan & Kesgin, 2007; Sen & Erdogan, 2014; Tüfekci et al., 2012).

5.4.1 Parameters

We experimented with different values for the number of TF-IDF features ranging from 1000 to 50000. TF-IDF features are extracted using *Gensim* toolkit (Řehůřek & Sojka, 2010). Vocabulary ordering is based on the number of occurrences of the words in the corpora. SVM model is implemented using the *scikit-learn* Python library (Pedregosa, Varoquaux, Gramfort, Michel, Thirion, Grisel, Blondel, Prettenhofer, Weiss, Dubourg, Vanderplas, Passos, Cournapeau, Brucher, Perrot

[6] https://github.com/ahmetax/trstop/blob/master/dosyalar/turkce-stop-words

[7] https://github.com/crodas/TextRank/blob/master/lib/TextRank/Stopword/turkish-stopwords.txt

[8] https://github.com/stopwords-iso/stopwords-tr/blob/master/stopwords-tr.txt

& Duchesnay, 2011) and we used the default penalty parameter (C). Results of TF-IDF and SVM modeling for various vocabulary sizes are given at Table-5.3

Table 5.3 Accuracies (%) of TF-TDF + SVM for Various Vocabulary Sizes

Corpus/Voc. Size	**1K**	**5K**	**10K**	**20K**	**50K**
Sabah	84,29	86,22	86,41	86,52	86.50
Cumhuriyet	69,12	71,71	71,81	71,72	71,69

We observe that a vocabulary size around 10.000 and 20.000 results in relatively better accuracies and increasing this size further does not yield performance improvement. With the light of these results, we used 10.000 for the vocabulary size of LDA training which is explained next.

Online Latent Dirichlet Allocation has a parameter τ that is used for changing the affect of first documents used in the training on the model and we used a τ value of 1024. We used a decay factor (κ) of 0.7, a batchsize of 100 and number of epochs was 3. Various values for the number of topics are used but they are all greater than or equal to the number of classes.

We used *Gensim* toolkit for unsupervised training of word embeddings (Řehůřek & Sojka, 2010). We used a context length of 20, and number of negative samples per example was 5, and number of epochs was 20. We tried 100, 200, 400 and 600 for the vector sizes. We discarded words that appear less than 10 times in the corpora from the dataset and the final vocabulary sizes are 70.118 for the Cumhuriyet corpus and 60.718 for the Sabah corpus.

We used a Neural Network with 2 hidden layers and 50 nodes at each hidden layer. We used ReLu activations for hidden layers and the sigmoid function for the output layer. Optimization is done by RMSprop learning algorithm and the learning rate is 0.01. Number of epochs is chosen to be 10, and the batchsize is chosen to be 100. Implementation is done by the *Pytorch* deep learning library (Paszke et al., 2017).

5.4.2 Results

Results are given at Table-5.4. LDA model which does not use any label resulted in the worst accuracies (purity score) as expected. Increasing the number of topics improved the performance for the Sabah corpus; however, for the Cumhuriyet corpus, best accuracy is obtained by the number of topics that is equal to the number of classes. We attribute this behaviour to the high number of classes for

the Cumhuriyet corpus and a number of classes of 4 for the Sabah corpus; so the variations in topics of a single class is expected to be much higher for the Sabah dataset.

For supervised models, TF-IDF with SVM and word embeddings with SVM models resulted in similar accuracies. However, increasing the embedding dimensions improves the performance and surpasses the performance of the TF-IDF model. This behaviour does not hold for the TF-IDF feature as increasing the feature size more than 20.000 did result in performance loss as shown in Table-5.3. This can be attributed to the phenomenon called "Curse of Dimensionality" and this result supports the benefit of low dimensional nature of word embeddings.

We used vocabulary sizes of approximately 70.000 and 60.000 for the NN models with word embeddings for the Sabah and the Cumhuriyet corpora respectively. Updating word embeddings through Back-Propagation resulted in higher accuracies than the SVM model with word embeddings and best results are obtained with these methods (%88.28 ve %74.31 for the Sabah and the Cumhuriyet datasets respectively). Another observation is that accuracies increase as embedding dimensions decreases and this shows that topic/category information can be embodied with very low dimensions of features using word embeddings and using traditional high dimensional vector-space representations such as TF-IDF would be cumbersome in terms of both accuracies and computational requirements.

Table 5.4 Accuracies (%). K values of the LDA are for the Sabah and the Cumhuriyet corpora respectively.

Method	**Sabah**	**Cumhuriyet**
LDA ($K = 4$ / $K = 14$)	65.41	47.94
LDA ($K = 10$ / $K = 20$)	67.60	43.31
LDA ($K = 20$ / $K = 30$)	72.08	45.37
TF-TDF ($10K$ Voc. size) + SVM	86.41	71.81
WE ($d = 100$) + SVM	85.47	70.34
WE ($d = 200$) + SVM	86.16	71.55
WE ($d = 400$) + SVM	86.72	72.24
WE ($d = 600$) + SVM	86.89	72.50
WE ($d = 100$) + NN	**88.28**	**74.31**
WE ($d = 200$) + NN	87.93	73.64
WE ($d = 400$) + NN	87.94	72.29
WE ($d = 600$) + NN	87.53	72.97

5.5 Conclusion

In this chapter, we introduced two new Turkish text corpora with category labels and publicly share them. We applied some text categorization models and reported the results. We observed that NN with average word embedding for document representations where embeddings are updated simultaneously with the Back-Propagation algorithm resulted in the best performance.

We believe that LDA, which does not use label information, has also promising results for the text categorization problem. This implicates that unsupervised or semi-supervised models should be worked on considering the increasing amount of text data and labeling large datasets is not as feasible, especially with the large volumes of new text data being produced daily in recent years.

CHAPTER 6

COMBINING LEXICAL AND SEMANTIC SIMILARITY METHODS FOR NEWS ARTICLE MATCHING

Matching news articles from multiple different sources with different narratives is a crucial step towards advanced processing of online news flow. Although, there are studies about finding duplicate or near-duplicate documents in several domains, none focus on grouping news texts based on their events or sources. A particular event can be narrated from very different perspectives with different words, concepts, and sentiment due to the different political views of publishers. We develop novel news document matching method which combines several different lexical matching scores with similarity scores based on semantic representations of documents and words. Our experimental result show that this method is highly successful in news matching. We also develop a supervised approach by labeling pairs of news documents as same or not, then extracting structural and temporal features. The classification model learned using these features, especially temporal ones and train a classification model. Our results show that supervised model can achieve higher performance and thus better suited for solving above mentioned difficulties of news matching. Some of the techniques developed in this work can be used in lexical analysis of a deception video. [1]

[1] Work at this chapter is presented at 2nd International Data Science Conference – iDSC2019 (Sen, Erdinc, Yavuzalp & Ganiz, 2019)

6.1 Introduction

The number of news from different sources and perspectives has dramatically increased due to the ever increasing variety of internet news portals and the rapid sharing of news on social media. A necessity for organizing and summarizing vast amounts of news items has emerged. Modelling news events from multiple sources would be useful for summarizing stories of long term event sequences as well as detecting false news. A necessary step of this news modelling, that we focus on in this work, is matching news articles from different portal sources that correspond to the same event. This is an important problem since a particular event can be narrated from very different perspectives with different words, concepts, and sentiment due to the different political views of publishers in a highly polarized society.

Although, there are studies about finding duplicate or near-duplicate documents in several domains, none focus on detecting same news texts which are expressed differently. News matching can be considered as a sub-problem of semantic similarity which aims to model the similarity of different textual elements considering the meaning of those elements (Guo, Fan, Ji & Cheng, 2019; Liu, Zhang, Niu, Lin, Lai & Xu, 2018). Semantic similarity is mostly studied for the information retrieval problems such as question answering, text summarization and web search. These problems, although numerous models have been proposed and applied to them (Guo et al., 2019; Hu, Lu, Li & Chen, 2014; Pang, Lan, Guo, Xu, Wan & Cheng, 2016), are fundamentally different than the news matching problem in two ways: First, at least one of the two items in the pair are short text documents, such as questions, queries, summaries, etc. However, news articles usually are longer. Second, unlike in semantic matching problems, queries are commutative in the news matching problem, i.e. both items in the query pair are news articles.

Despite various architectures and models on different semantic similarity problems, matching long text documents is still a challenging problem (Liu et al., 2018).

We approach the problem from several aspects. First, we investigate the performance of simple lexical matching scores for the news matching problem. We argue that, even though two matched news articles have different narratives, the number of keywords and entities that define the event of concern is reasonably high for long articles and that is in favor for the lexical matching based scoring methods. We experiment on various lexical matching methods, and propose a value for threshold parameter. Following this we show an improved performance using semantic similarity which leverage word embeddings. In addition to this, combining lexical

matching scores and cosine similarity scores of different word embedding methods, namely Word2Vec (Mikolov et al., 2013b) and FastText (Bojanowski, Grave, Joulin & Mikolov, 2016,1; Joulin, Grave, Bojanowski & Mikolov, 2016), improves the performance further. These were unsupervised methods.

As mentioned before, some matched news articles may have different level of details about an event. This imbalance creates noise for the lexical matching scores. To alleviate this problem, we obtain additional scores between other fields of the two news articles, namely title and spot. We show improved performance using these additional field scores.

We also develop with supervised classification models using similarity scores as features along with other features such as the time difference and length difference of news articles. These supervised models seems especially effective.

In Section-6.2 we summarize some previous work. In Section-6.3 we define the problem, explain the unsupervised and supervised methods that we applied. In Section-6.4 we first describe the dataset that we collected, define the experimental setup, present results and discuss them. We give concluding remarks and future work at Section-6.6. [2]

6.2 Related Work

As discussed at Section-6.1, most of the existing works on semantic similarity focus on problems such as question answering, text summarization and search (Guo et al., 2019). Traditional methods exploit lexical databases such as WordNet (Corley & Mihalcea, 2005), or any other structured semantic knowledge sources as for the biomedical text matching problem (McInnes & Pedersen, 2013; Pedersen, Pakhomov, Patwardhan & Chute, 2007), to measure the semantic similarity of words. However, using such external lexical resources is not practically applicable to our problem, because technical terms and named entities in the news items are of vast domain and evolve in time. In addition, high quality WordNet databases are not available in all languages.

More recent methods of semantic similarity use word embeddings that are obtained by unsupervised training on large corpora. Kenter et al. apply BM25 algorithm

[2]The dataset and codes are available at: `https://github.com/donanimhaber/newsmatching`

(Robertson, Zaragoza & others, 2009) to word embeddings, along with other meta-features, for semantic similarity of short texts (Kenter & De Rijke, 2015). We compare with this algorithm in our experiments. Kusner et al. introduce Word Mover's Distance (MVD) which casts the dissimilarity of two sets of word embeddings as Earth Mover's Distance transportation problem, where each word embedding from one document moves to embeddings in the other document with some proportions and the minimum distance is considered as the dissimilarity (Kusner, Sun, Kolkin & Weinberger, 2015). They report that removing one constraint in the optimization problem results in slightly lower performance but the computation time is highly reduced. We compare with this similarity algorithm, which they call Relaxed WMD (RWMD), in our experiments.

Recent supervised models mostly employ Deep Neural Network (DNN) architectures to achieve better accuracy on these tasks than the traditional methods (Hu et al., 2014; Pang et al., 2016; Yin, Schütze, Xiang & Zhou, 2016). Pang et al. obtain a matrix with word embeddings that contains interactions of word pairs among the two documents and treat the matrix as an image to feed into a Convolutional Neural Network (CNN) with dynamic pooling layer (Pang et al., 2016). Hu et al. uses a similar architecture with max pooling and apply their model to various Natural Language Processing (NLP) problems such as sentence completion, matching a response to a tweet, etc. (Hu et al., 2014). The only work, to the best of our knowledge, that deals with matching long texts of news events is the work of Liu et al. (Liu et al., 2018). For the purpose of efficient learning with Neural Networks for long documents, they first embed these long documents into a *concept* graph, where nodes in the graph represent different concepts. Each sentence in the document is assigned to a different concept. Then, graph pairs are fed into a *Siamese Encoded Graph CNN*. They obtain better results for the news matching problem than the general similarity matching models.

These DNN models perform poorly in our problem because they require large amounts of labeled data, whereas in our dataset a small ($\approx 2K$) number of news articles are labeled. Our trials with smaller DNN models performed poorly on our dataset, therefore we discard these results from the Experiments Section.

6.3 Method

6.3.1 Problem Definition

We tackle the problem of matching news with the same or very similar content of different portal sources. Document matches are postulated as news stories depicting the same event. We formulate the problem as inputting a pair of documents to the model and outputting a binary decision as "same" or "different".

Each document has the following fields:

1.1 Title: Heading of the news story.

1.2 Spot: Sub-heading of the story.

1.3 Body: Details and the main content of the story.

1.4 Date-time: Last modified date and time of the story.

We assume that some fields can be empty for some documents. We created another field named "text" that is the concatenation of "title", "spot" and "body" fields and used this field instead of "body". So "text" field is all the textual content of the document, and we did not use the "body" since it is usually very close to "text" because of the short lengths of "title" and "spot".

Textual fields (title_i, spot_i and text_i) of a document doc_i are sequences of words, for example $S^{(i)}_{\text{title}} = [w_0, w_1, \ldots, w_N]$.

First we describe our text based methodologies for unsupervised scoring of document pairs. In the following section, we describe the supervised setting and methods.

6.3.2 Unsupervised Scoring

Given two documents, doc_1 and doc_2, we calculate four different similarity scores for all three fields "title", "spot" and "text". Three of these scores are based on lexical matching. Another scoring uses word embeddings to capture semantic similarity.

6.3.2.1 Lexical Matching Scores

Jaccard Similarity Coefficient (JSC) (Pandit, Gupta & others, 2011) with unique words in the field is obtained as follows:

$$\text{JU}(\text{field}_i, \text{field}_j) = \frac{\left|U^{(i)}_{\text{field}} \cap U^{(j)}_{\text{field}}\right|}{\left|U^{(i)}_{\text{field}} \cup U^{(j)}_{\text{field}}\right|}, \tag{6.1}$$

where $U^{(i)}_{\text{field}}$ is the set of unique words in the sequence $S^{(i)}_{\text{field}}$, $|A|$ is the cardinality of a set A, $\cap$ is the intersection operator and $\cup$ is the union operator.

We calculate JSC also with all words as opposed to unique words:

$$\text{JC}(\text{field}_i, \text{field}_j) = \frac{\left|C^{(i)}_{\text{field}} \cap C^{(j)}_{\text{field}}\right|}{\left|C^{(i)}_{\text{field}} \cup C^{(j)}_{\text{field}}\right|}, \tag{6.2}$$

where $C^{(j)}_{\text{field}}$ is obtained by enumerating words by the occurrence count in the field. For example, for a field with words $S_{field} = [\text{“the”}, \text{“cat”}, \text{“sat”}, \text{“on”}, \text{“the”}, \text{“mat”}]$, the counted field set is $S_{field} = \{\text{“the-1”}, \text{“cat-1”}, \text{“sat-1”}, \text{“on-1”}, \text{“the-2”}, \text{“mat-1”}\}$ at which the second occurrence of the word “the“ is now different than the first occurrence.

One issue we observed with these scorings is that for news pairs that give substantially different amount of details about an event we obtain poor JU and JC scores. Therefore we define another scoring function that replaces the denominator of the JU function with the length of the short document field:

$$\text{JS}(\text{field}_i, \text{field}_j) = \frac{\left|U^{(i)}_{\text{field}} \cap U^{(j)}_{\text{field}}\right|}{\min\left(\left|U^{(i)}_{\text{field}}\right|, \left|U^{(j)}_{\text{field}}\right|\right)}, \tag{6.3}$$

Even though we expect that JS would result in better similarity scores for hand matched pairs with a large difference in length, it may result in poor performance on other pairing cases, resulting in lower performance on overall dataset. But, we

keep this scoring to be used as a feature with supervised classification models.

6.3.2.2 Word Embedding Scores

We use two different word embedding models: Word2Vec and FastText.

Word2Vec is a continuous word representation where each word is mapped to a low dimensional real vector (Mikolov et al., 2013b). We use the Skip-gram model which learns from large textual corpus with the objective of predicting the context words, i.e. nearby words, given an input word. A large unlabeled news articles corpus is used to train the model. After training, we obtain the vector of a given document by averaging vectors of the words contained in the document.

FastText is another continuous word representation method. Unlike previous unsupervised word embedding models such as Word2Vec (Mikolov et al., 2013b), Doc2Vec (Le & Mikolov, 2014), Glove (Pennington et al., 2014); FastText learns embeddings of character n-grams in addition to the word n-grams using skip-gram model (Bojanowski, Grave, Joulin & Mikolov, 2017b). It obtains the representation of a word by averaging over character n-gram vectors and the word vector. For example vector of the word "the" is obtained by averaging over the word vector "the" and the character n-gram vectors of "<t", "th", "he", "e>", "<th", "the", "he>" if only 2-grams and 3-grams are used.

Using character n-gram embeddings paves the way to dumping syntactic information into the representative vectors. This may particularly beneficial for morphologically rich languages such as Turkish, Finnish, and Hungarian. In addition, it helps to deal with syntax errors and typos which occurs frequently in non-editorial texts.

Since the number of n-grams would be enormous for a reasonably large corpus and a reasonable selection of n in the n-gram such as 3 to 6; memory requirement of this algorithm would be very high. Mikolov et al. deal with this problem by randomly grouping n-grams using a hash function and using the same vector for n-grams that are in the same group (Bojanowski et al., 2017b).

FastText obtains the document vector by first calculating word vectors, normalizing them such that their l_2 norm is 1 and then averaging them.

We use the cosine similarity to obtain the word embedding score of a pair of docu-

ment fields:

$$\text{WE}(\text{field}_i, \text{field}_j) = \frac{\mathbf{v}_i^T \mathbf{v}_j}{\|\mathbf{v}_i\| \, \|\mathbf{v}_j\|}, \tag{6.4}$$

where $\mathbf{v}_i$ is the FastText (FT) or Word2Vec (WV) vector of field_i and $\|.\|$ is the l_2 norm.

6.3.2.3 Thresholding

We obtain an optimal threshold for each scoring function and for each field using a labeled dataset of news articles. We use the threshold for which precision is equal to recall since the number of negative examples, i.e. pairs that do not match, is much higher than the number of positive examples. This corresponds to the threshold for which the number of *false negatives* (falsely classified as negatives) is equal to *false positives* (falsely classified as positives).

6.3.2.4 Combination with Weighted Averaging

We combine different scores by weighted averaging. First, we normalize each score using the corresponding threshold and the standard deviation:

$$\text{JU}_{\text{norm}}(\text{field}_i, \text{field}_j) = \frac{\text{JU}(\text{field}_i, \text{field}_j) - \text{thr}_{\text{JU-field}}}{\text{std}(\{\text{JU}(\text{field}_i, \text{field}_j)\}_{(i,j)\in X})} \tag{6.5}$$

where X contains the index pairs in the training data, $\text{thr}_{\text{JU-field}}$ is the optimal threshold for the method JU and the field *field*. Normalized scores for other fields are computed similarly. After normalization of individual scores, we obtain the

combined score as follows:

$$\text{COMB}(\text{field}_i, \text{field}_j) = \frac{1}{5} \sum_{m \in \{\text{JU,JS,JC,FT,WV}\}} \sum_{f \in \{\text{title,spot,text}\}} w_f s_{fm}^{(i,j)} \tag{6.6}$$

where w_f is the weight of field f independent of the scoring method, $s_{fm}^{(i,j)}$ is the normalized score of documents i and j with the method m for the field f, such as $\text{JU}_{\text{norm}}(\text{title}_i, \text{title}_j)$ for field *title* and method *JU*. We choose the weights manually proportional to the average lengths of the fields in the training data, i.e. $w_{\text{title}} < w_{\text{spot}} < w_{\text{text}}$ and weights sum to one.

Optimal threshold for the combined score is obtained similar to those of previous scoring methods. However, since the normalized scores have the optimal threshold of 0, the resulting optimal threshold for the combination is very close to 0.

6.3.2.5 Comparision of Semantic Similarity Methods

We compare with two similarity scoring methods which employ word embeddings. First one is the Word Mover's Distance (WMD) (Kusner et al., 2015) which minimizes the following constrained optimization problem:

$$\min_{\mathbf{T} \geq 0} \sum_{i,j} T_{i,j} c(i,j) \tag{6.7}$$

subject to

$$\sum_j T_{ij} = d_i \quad \forall i \tag{6.8}$$

$$\sum_i T_{ij} = d'_j \quad \forall j \tag{6.9}$$

where, $c(i,j)$ is the Euclidean Distance between embeddings of the i^{th} and j^{th} unique words of the first and second document respectively, d_i and d'_j are the frequencies of the corresponding words in the document. After finding optimal $\mathbf{T}$, the distance between documents is computed by $\sum_{i,j} T_{i,j} c(i,j)$. Since solving above problem is computationally difficult, Kusner et al. relaxed this optimization problem by

removing one of the constraints, obtain two easy-to-optimize problems one for each constraint (Relaxed WMD). Maximum of the two distances is used as the final distance. They show that RWMD obtains similar classification results with the WMD, therefore we obtained results using RWMD.

Kenter et al. (Kenter & De Rijke, 2015) uses the following BM25 similarity algorithm (Robertson et al., 2009) for short texts of S_1 and S_2:

$$f(S_1, S_2) = \sum_{w \in S_1} \text{IDF}(w) \frac{\text{sem}(w, S_2)(k_1 + 1)}{\text{sem}(w, s) + k_1(1 - b + b\frac{|S_2|}{L})} \tag{6.10}$$

where, IDF(w) is the Inverse Document Frequency, L is the average document length, $|S_1| \geq |S_2|$, b and k_1 are meta-parameters and sem is defined as follows:

$$\text{sem}(w, S) = \max_{w' \in S} \frac{\mathbf{v}_w^T \mathbf{v}_{w'}}{||\mathbf{v}_w^T||\,||\mathbf{x}_{w'}||} \tag{6.11}$$

where, $\mathbf{v}_w$ is the word embedding of the word w.

6.3.3 Supervised Classification

We use the scores described in previous section as features to be fed into a classifier. In addition, we extract *length* features and *time* features. For *length* features, we use mainly two inputs: l_1 and l_2 which represent the lengths (number of words) of fields of the document pair. We extract the following *length* features for each field ("title", "spot", "text"):

- Minimum of lengths: $\min(l_1, l_2)$
- Maximum of lengths: $\max(l_1, l_2)$
- Absolute value of difference of lengths: $|l_1 - l_2|$
- Absolute value of the difference divided by maximum of lengths $|l_1 - l_2| / \max(l_1, l_2)$
- Maximum length divided by the minimum length: $\max(l_1, l_2) / \min(l_1, l_2)$

In addition to the *text length* features, we extract *time* features which are the difference of the last modified times of the two news articles. We extract two features corresponding to time difference in hours and in days. These features provide significant information to the classifier since news articles are published mostly on the same day with the event of subject.

We have 16 score features, 15 textual length features and 2 time features which amounts to a total of 33 features. These features are then fed to various classifiers including *Random Forest* (RF), *Support Vector Machines* (SVM) and *Multilayer Perceptron* (MLP). Majority Voter (MV) classifier combination results are also reported.

6.4 Experiments

In subsequent sections we describe the dataset, preprocessing of the texts, parameters and details of the implementation and give results and discuss them.

6.4.1 Labeled News Dataset

We evaluate the methods on a news articles corpus in Turkish whose URLs are obtained manually by searching for the articles of the same events on different news portals. Then we crawled the web pages to obtain the fields "title", "spot", "body" and "date-time".

There are in total 20 different news portals and 2049 news items in the dataset. News articles span approximately 6 months, but majority of the articles are in a 1 month period. We obtained 693 groups of news where news in the same group correspond to the same event, i.e. positively labeled. Each group contains 2.75 documents on average. We obtained a total of 1858 positive examples and randomly choose $15,000$ negative examples. Our dataset has a total of 16858 news texts. Average numbers of words in the dataset are 6.94 ± 2.82, 25.72 ± 13.86 and 205.4 ± 223.72 for the fields *title*, *spot* and *body* respectively where second arguments are the standard deviations.

We also use an unlabeled news corpus in Turkish of size ≈ 4.7 million news texts for training the Word2Vec and FastText models. This unlabeled news corpus does not contain news texts of the labeled dataset. We apply the same preprocessing steps on the unlabeled as with the labeled dataset.

6.4.2 Preprocessing

We apply the following preprocessing steps to all text fields:

- Escape *html* character references.
- Remove *html* tags.
- Lowercase.
- Sentence tokenizer using NLTK toolkit (Bird, Loper & Klein, 2009).
- Lemmatization - Morphological analysis and disambiguation with Zemberek toolkit (Akın & Akın, 2007) to get lemmas.
- Stopword removal.

6.4.3 Settings

We use vector dimension of 100 for all word embeddings methods. For Word2Vec, gensim toolkit is used with skip-gram model and negative sampling (Řehůřek & Sojka, 2010). Context window length parameter is chosen as 5, minimum count for filtering out words is set to 5 and training performed for 5 epochs. There were no improvement on the loss value after a few epochs.

For the FastText model; minimum count for filtering out words is set to 5, context window length parameter is chosen as 5, bucket size is chosen to be $2,000,000$ and from 3-grams up to (including) 6-grams are used for character n-gram embeddings. Training performed for 100 epochs.

For combination of different fields and lexical matching methods as in (6.6), we used the following weights: $w_{\text{title}} = 0.1$, $w_{\text{spot}} = 0.3$, $w_{\text{text}} = 0.6$.

Scores for missing fields are set to the mean of the corresponding field's scores in the dataset.

We used Random Forest (RF), Multilayer Perceptron (MLP) and Support Vector Machines (SVM) for the supervised classification. Models are implemented using the *sklearn* toolkit (Pedregosa et al., 2011). For the RF, 100 trees are used with 2 as the minimum samples per split, 2 as the minimum samples per leaf, and Gini impurity as the split criterion. For MLP, we used 2 layers with 500 nodes each, *adam* solver, batchsize of 100 and Relu activations. Training is stopped when the loss is not decreased for at least $1e-4$ in 10 epochs. For the SVM classifier, we used *RBF* kernel and applied grid search for the penalty parameter (C) and the RBF kernel parameter (γ). We searched in $\{1e-7, 1e-5, 1e-3, 1e-1, 1, 10\}$ and $\{1, 10, 100, 1000, 10000, 100000\}$ for γ and C respectively.

We normalized all the classifier features to the range $[0, 1]$. 10-fold Cross Validation (CV) are applied to test the performances. For the feature that divides maximum length by the minimum, we used the maximum of the feature along the dataset if one of the field is missing and used 1 if both fields are missing.

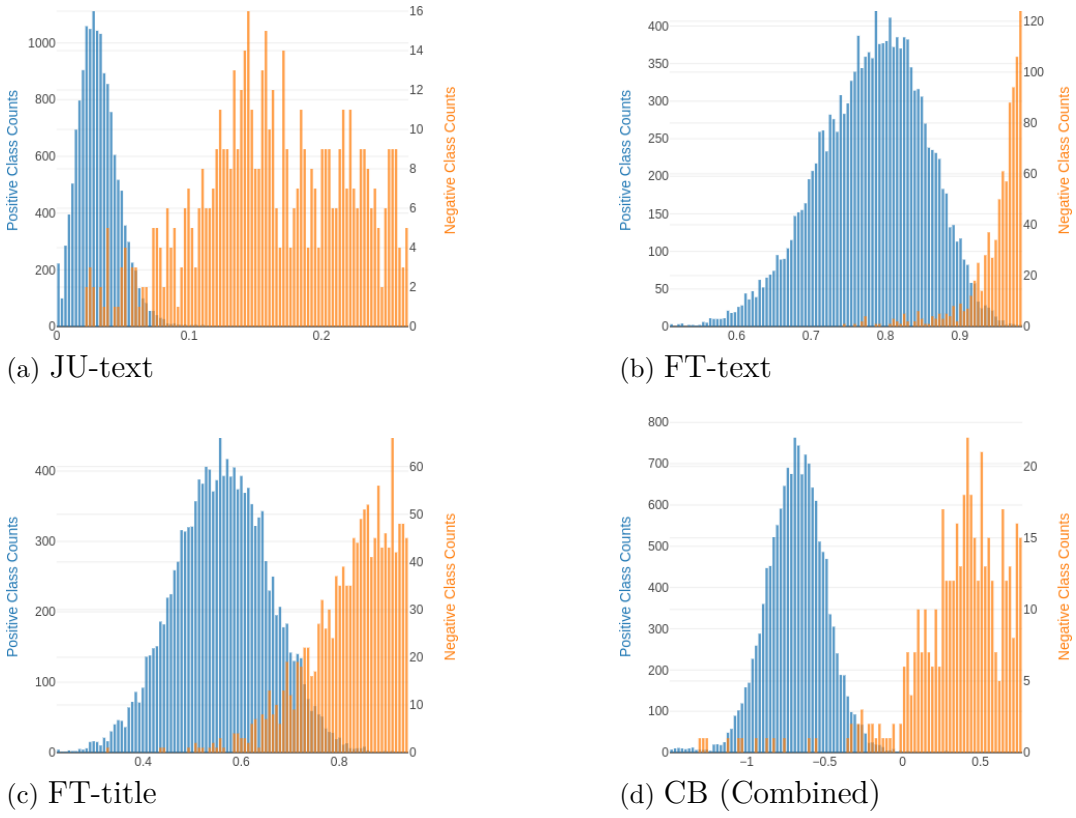
For the BM25 algorithm, we applied grid search for the meta-parameters b and k_1 in $\{0.8, 1, 1.2, 1.5, 2, 2.5, 5, 10, 20, 50\}$ and $\{0, 0.001, 0.01, 0.1,$
$0.2, 0.5, 0.75, 0.9, 1\}$ respectively. For the RWMD method, we calculated Inverse Document Frequencies (IDF) from the large unlabeled news corpora.

6.5 Results

Results for lexical matching based and word embedding based similarity scoring along with compared algorithms are shown at Table-6.1. Here, *FT* stands for FastText, *WV* stands for Word2Vec, *COMB.* is the average of different scores as in (6.6).

We obtained scores for all fields *title*, *spot*, *text* along with the combination of scores for these three fields, which is depicted with *Weighted Av.* in the table, similar to using (6.6) except with only the related method. *Combined* results are also computed similarly.

Our results show that in general lexical matching works better than word embedding based similarity methods, except for the *title* field for which FastText works better.



(a) JU-text (b) FT-text

(c) FT-title (d) CB (Combined)

Figure 6.1 Score histograms of negative and positive pairs for some methods and fields

Table 6.1 F1 Results of Unsupervised Methods for Different Fields

Method	**title**	**spot**	**text**	**Weighted Av.**
RWMD-FT	0.8299	0.9386	0.9263	0.9645
RWMD-WV	0.8465	0.8762	0.9440	0.9758
BM25-FT	0.7438	0.8665	0.8881	0.9333
BM25-WV	0.7508	0.8741	0.8994	0.9413
Cosine FT	0.8407	0.9182	0.9273	0.9537
Cosine WV	0.8423	0.9225	0.9177	0.9403
JU	0.8328	**0.9535**	0.9639	0.9833
JS	0.8280	0.9476	0.9459	0.9839
JC	0.8339	0.9533	0.9709	0.9839
COMB. (FT,WM)	0.8466	0.9241	0.9225	0.9499
COMB. (JU,JS,JC)	0.8341	0.9532	**0.9817**	**0.9887**
COMB. (ALL)	**0.8617**	0.9532	0.9726	0.9833

This shows that as the length of the text decreases, importance of semantic information increases for text matching. Another useful observations is that FastText achieves higher performance than the Word2Vec model.

Compared algorithm RWMD outperforms cosine similarity of word embeddings, but performs worse than the lexical matching based results. BM25 algorithm does not work well even though extra IDF information is incorporated.

Even though *title* and *spot* fields result in lower scores than the *text* field, score level combination of three fields (*all* in the table) achieves higher performance than the *text* field itself, even though *text* field already contains the *title* and *spot* in its content. This is expected since some news pairs have a high difference in the amount of details and titles or spots may result in noise-free match.

Among different lexical matching methods (*JU*, *JS*, *JC*), *JU* performs the best although results are close to each other. Results for score level combination are depicted with *COMB.* at the table. Best performing method is the score level combination of lexical matching methods, i.e. *JU*, *JS* and *JC* with the weighted averaging of fields. However, word embedding based combination works better for the *title* field.

We present histograms of scores of negative and positive pairs for some methods and fields at Figure-6.1. Note that we use the right y-axis for positive class for clear visualization since the number of positive examples are much lower in the dataset. We observe that lexical matching based methods result in closer to uniform histograms than word embedding based methods. However, the combined method yields more Gaussian like distribution for the positive examples. We also see higher interference between the positive and negative classes at the *title* histogram than the *text* histogram of FT.

Table 6.2 Results for Supervised Methods

Method	**F1**	**Prec.**	**Recall**	**Acc.**
WE-MLP	0.9895	0.9919	0.9871	0.9977
WE-RF	0.9879	0.9876	0.9882	0.9973
WE-SVM	0.9922	0.9935	0.9909	0.9983
WE-MV	0.9922	0.9925	0.9919	0.9983
MLP	0.9925	0.9935	0.9914	0.9983
RF	0.9911	0.9914	0.9909	0.9980
SVM	0.9919	0.9935	0.9903	0.9982
MV	**0.9930**	**0.9941**	**0.9919**	**0.9985**

Results of supervised classifications are depicted at Table-6.2. We experimented using only word-embedding based similarity scores (FT and WV) along with addi-

tional features to test the benefit of lexical matching based scores in the supervised setting. We see improvements for all of the classifiers which shows the benefit of lexical matching scoring for news matching.

All supervised methods result in better performance than the best unsupervised score thresholding method (depicted as *COMB. (all)* at Table-6.1).

The final F1 score is 0.9930 which corresponds to falsely classifying 26 pairs in total. Some of these errors are pairs with the same narratives but for different events, such as lottery results, weather reports, etc. Another misclassification pattern is the huge difference in details between two news articles, such as a document with one sentence v.s. document with more than 100 sentences. In these cases, title matching scores are not high enough to balance the text matching scores.

6.6 Conclusion

In this chapter, we propose novel lexical matching based similarity calculation methods for matching long text articles of same events with different narratives. Since long articles contain higher number of event related keywords and entities than short text documents such as queries or questions, we show that, lexical matching based scores are able to obtain a fine discrimination between matched and unmatched pairings, even without the use of labels. We also proposed that obtaining scores for different fields such as title and spot of the news article would make the model more robust. Using these scores as features to be fed to classifiers, together with other length and time based features, improved the performance even further.

As a future work, we plan to work on matching two sets of news as opposed to single news, so that we can benefit from previous matchings. Another effort would be online learning of the model and testing the system in real-time.

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

In Chapter 2, we presented our work on multimodal deception detection evaluated our model both qualitatively and quantitatively on a real-life high-stakes dataset. The dataset contains videos of subjects from public trials. We convert the dataset to a subject-level setting rather than a video-level setting and we argue that this setting is more appropriate to the deception detection problem. We extracted features from verbal, acoustic and visual modalities and built classifiers using them. We propose to use new acoustic features, namely f0-mean, f0-std and speech/silence length histograms. Our results indicate that, combining different modalities at the score level results in an accuracy of 84.18%. We also analyzed the effects of individual visual gestures and word usages on deception decision. We reported that these cues of deception are in line with the deception detection literature.

In Chapter 3, we introduced a novel approach for single channel source separation using a deep neural network for learning the manifolds of sources. The network is trained discriminatevly for classifying the source type given the utterance using the training data of individual sources. In the separation phase, the trained DNN is used in an energy minimization framework for enforcing each estimated source signal to reside in their corresponding manifold while additional loss functions ensure that the weighted sum of the sources is close to the input mixed signal. We initialize the source estimates with the predictions of the NMF model and fine-tune them using the DNN model. Results of our experiments show that the predictions of the DNN

model improves the predictions of the NMF model.

In Chapter 4, we trained word embeddings using the skip-gram model with a large Turkish corpus. We created question sets for measuring the semantic and syntactic qualities of word embeddings. We believe this work had a good impact in the Turkish NLP community in terms of usage of unsupervised training of word embeddings, which are frequently and successfully used in English, for Turkish. To the best of our knowledge, this is the first work of applying the skip-gram models to Turkish. We compare models with different embedding dimensions. In addition, we compare hierarchical-maximum method with the negative sampling method and report that negative-sampling works better. We also report that removing suffixes from the dataset improves the performances.

In Chapter 5, we introduced two new Turkish text corpora of online news articles and publicly share them. We automatically extracted the article categories and cast them as labels. We applied some text categorization models and reported the results. We observed that NN with average word embedding for document representations where embeddings are updated simultaneously with the Back-Propagation algorithm resulted in the best performance. We also report the results of the LDA model, which is an unsupervised topic model, to the categorization problem and report that results are promising.

In Chapter 6, we proposed novel lexical matching based similarity calculation methods for matching long text articles of same events with different narratives. Since long articles contain higher number of event related keywords and entities than short text documents such as queries or questions, we show that, lexical matching based scores are able to obtain a fine discrimination between matched and unmatched pairings, even without the use of labels. We also proposed that obtaining scores for different fields such as title and spot of the news article would make the model more robust. Using these scores as features to be fed to classifiers, together with other length and time based features, improved the performance even further.

7.2 Future Work

In the future, we will work on improving automatic gesture identification and automatic speech transcription, with the goal of taking steps towards a real-time deception detection system. We believe a critical issue of improving performances

deception detection systems is the lack of large real-life deception datasets. In this regard, we believe that an important contribution to the problem would be enlarging the available data. Another possible future direction would be extracting temporal features or using tempral models for incorporating the time-dependent information into to the model.

Many adjustments for the model parameters can be done to improve the proposed single channel source separation model introduced in Chapter 3. Different types of DNN such as deep autoencoders and deep recurrent neural networks which can handle the temporal structure of the source signals can be tested on the SCSS problem. We believe our idea is a novel idea and many improvements will be possible in the near future to improve its performance.

In Chapter 4, we trained word embeddings for Turkish. We believe, unsupervised models that take into account the rich morphological structure of the Turkish language would have a high impact on various NLP problems.

In Chapter 5, we showed that the unsupervised topic model-LDA has promising results for the text categorization problem. This implicates that unsupervised or semi-supervised models should be worked on considering the increasing amount of text data and labeling large datasets is not as feasible, especially with the large volumes of new text data being produced daily in recent years. Another possible future direction would be employing attention mechanisms with the neural networks instead of averaging word embeddings of documents.

As a future work for the semantic matching model proposed in Chapter 6, we plan to work on matching two sets of news as opposed to single news, so that we can benefit from previous matchings. Another effort would be online learning of the model and testing the system in real-time.

BIBLIOGRAPHY

Aamodt, M. & Custer, H. (2006). Who can best catch a liar? a meta-analysis of individual differences in detecting deception. *Forensic Examiner*, *15*(1), 6–11.

Abouelenien, M., Pérez-Rosas, V., Mihalcea, R., & Burzo, M. (2014). Deception detection using a multimodal approach. In *Proceedings of the 16th International Conference on Multimodal Interaction*, ICMI '14, (pp. 58–65)., Istanbul, Turkey. ACM.

Abouelenien, M., Pérez-Rosas, V., Mihalcea, R., & Burzo, M. (2016). Detecting deceptive behavior via integration of discriminative features from multiple modalities. *IEEE Transactions on Information Forensics and Security*, *12*(5), 1042–1055.

Akin, A. A. & , M. D. A. (2007). Zemberek, an open source nlp framework for turkic languages. Yeni versiyon: https://github.com/ahmetaa/zemberek-nlp.

Akın, A. A. & Akın, M. D. (2007). Zemberek, an open source nlp framework for turkic languages. *Structure*, *10*, 1–5. `https://github.com/ahmetaa/zemberek-nlp`.

Allwood, J., Cerrato, L., Jokinen, K., Navarretta, C., & Paggio, P. (2007). The mumin coding scheme for the annotation of feedback, turn management and sequencing phenomena. *Language Resources and Evaluation*, *41*(3-4), 273–287.

Almela, A., Valencia-García, R., & Cantos, P. (2012). Seeing through deception: A computational approach to deceit detection in written communication. In *Proceedings of the Workshop on Computational Approaches to Deception Detection*, (pp. 15–22)., Avignon, France. Association for Computational Linguistics.

Ay, S., Doğan, Y. S., Alver, S., & Kaya, Ç. (2016). A novel attribute weighting method with genetic algorithm for document classification. In *Signal Processing and Communication Application Conference (SIU), 2016 24th*, (pp. 1129–1132). IEEE.

Bachenko, J., Fitzpatrick, E., & Schonwetter, M. (2008). Verification and implementation of language-based deception indicators in civil and criminal narratives. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, (pp. 41–48).

Baltrušaitis, T., Robinson, P., & Morency, L.-P. (2016). Openface: an open source facial behavior analysis toolkit. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, (pp. 1–10). IEEE.

Baltrusaitis, T., Zadeh, A., Lim, Y. C., & Morency, L.-P. (2018). Openface 2.0: Facial behavior analysis toolkit. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, (pp. 59–66). IEEE.

Bartlett, M., Littlewort, G., Frank, M., Lainscsek, C., Fasel, I., & Movellan, J. (2006). Automatic recognition of facial actions in spontaneous expressions. *Journal of Multimedia*, *1*(6), 22–35.

Bengio, Y., Ducharme, R., Vincent, P., & Janvin, C. (2003). A neural probabilistic language model. *J. Mach. Learn. Res.*, *3*, 1137–1155.

Bird, S., Loper, E., & Klein, E. (2009). *Natural language processing with python*

O'reilly media Inc.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, *3*(Jan), 993–1022.

Bloomfield, R. (2012). Discussion of detecting deceptive discussions in conference calls. *Journal of Accounting Research*, *50*(2), 541–552.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016). Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, *5*, 135–146.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017a). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135–146.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017b). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135–146.

Braun, M. T., Van Swol, L. M., & Vang, L. (2015). His lips are moving: Pinocchio effect and other lexical indicators of political deceptions. *Discourse Processes*, *52*(1), 1–20.

Burgoon, J., Mayew, W. J., Giboney, J. S., Elkins, A. C., Moffitt, K., Dorn, B., Byrd, M., & Spitzley, L. (2016). Which spoken language markers identify deception in high-stakes settings? evidence from earnings conference calls. *Journal of Language and Social Psychology*, *35*(2), 123–157.

Burgoon, J., Twitchell, D., Jensen, M., Meservy, T., Adkins, M., Kruse, J., Deokar, A., Tsechpenakis, G., Lu, S., Metaxas, D., Nunamaker, J., & Younger, R. (2009). Detecting concealment of intent in transportation screening: A proof of concept. *IEEE Transactions on Intelligent Transportation Systems*, *10*(1), 103–112.

Burns, M. B. & Moffitt, K. C. (2014). Automated deception detection of 911 call transcripts. *Security Informatics*, *3*(1), 8.

Caso, L., Maricchiolo, F., Bonaiuto, M., Vrij, A., & Mann, S. (2006). The impact of deception and suspicion on different hand movements. *Journal of Nonverbal Behavior*, *30*(1), 1–19.

Cataltepe, Z., Turan, Y., & Kesgin, F. (2007). Turkish document classification using shorter roots. In *Signal Processing and Communications Applications, 2007. SIU 2007. IEEE 15th*, (pp. 1–4). IEEE.

Chittaranjan, G. & Hung, H. (2010). Are you awerewolf? detecting deceptive roles and outcomes in a conversational role-playing game. In *2010 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, (pp. 5334–5337).

Cohen, D., Beattie, G., & Shovelton, H. (2010). Nonverbal indicators of deception: How iconic gestures reveal thoughts that cannot be suppressed. *Semiotica*, *2010*(182), 133–174.

Collobert, R. & Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, (pp. 160–167)., New York, NY, USA. ACM.

Corley, C. & Mihalcea, R. (2005). Measuring the semantic similarity of texts. In *Proceedings of the ACL workshop on empirical modeling of semantic equivalence and entailment*, (pp. 13–18). Association for Computational Linguistics.

Deoras, A. N. & Hasegawa-Johnson, A. (2004). A factorial hmm approach to simul-

taneous recognition of isolated digits spoken by multiple talkers on one audio channel. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, (pp. I–861). IEEE.

Depaulo, B., Malone, B., Lindsay, J., Muhlenbruck, L., Charlton, K., & Cooper, H. (2003). Cues to deception. *Psychological Bulletin*, *2003*, 74–118.

Derksen, M. (2012). Control and resistance in the psychology of lying. *Theory and Psychology*, *22*(2), 196–212.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv e-prints*, *2018*, arXiv:1810.04805.

Ekman, P. (2001). *Telling Lies: Clues to Deceit in the Marketplace, Politics and Marriage*. Norton, W.W. and Company.

Ekman, P., Friesen, W. V., & Hager, J. C. (2002). Facial action coding system: The manual on cd rom. *A Human Face, Salt Lake City*, *2002*, 77–254.

Ekman, P. & Rosenberg, E. (2005). *What the Face Reveals: Basic and Applied Studies of Spontaneous Expression Using the Facial Action Coding System (FACS)*. Series in Affective Science. Oxford University Press.

Eyben, F., Weninger, F., Gross, F., & Schuller, B. (2013). Recent developments in opensmile, the munich open-source multimedia feature extractor. In *Proceedings of the 21st ACM international conference on Multimedia*, (pp. 835–838). ACM.

Feng, S., Banerjee, R., & Choi, Y. (2012). Syntactic stylometry for deception detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12, (pp. 171–175)., Stroudsburg, PA, USA. Association for Computational Linguistics.

Févotte, C., Bertin, N., & Durrieu, J.-L. (2009). Nonnegative matrix factorization with the itakura-saito divergence: With application to music analysis. *Neural computation*, *21*(3), 793–830.

Fornaciari, T. & Poesio, M. (2013). Automatic deception detection in Italian court cases. *Artificial Intelligence and Law*, *21*(3), 303–340.

Fuller, C. M., Biros, D. P., Burgoon, J., & Nunamaker, J. (2013). An examination and validation of linguistic constructs for studying high-stakes deception. *Group Decision and Negotiation*, *22*(1), 117–134.

Gannon, T., Beech, A., & Ward, T. (2009). *Risk Assessment and the Polygraph*, (pp. 129–154). John Wiley and Sons Ltd.

Goldberg, Y. (2016). A primer on neural network models for natural language processing. *J. Artif. Intell. Res.(JAIR)*, *57*, 345–420.

Google (2019). Cloud speech-to-text recognition. `https://cloud.google.com/speech-to-text/`. Accessed: 2019-10-14.

Grais, E. M. & Erdogan, H. (2011a). Single channel speech music separation using nonnegative matrix factorization and spectral masks. In *2011 17th International Conference on Digital Signal Processing (DSP)*, (pp. 1–6). IEEE.

Grais, E. M. & Erdogan, H. (2011b). Single channel speech music separation using nonnegative matrix factorization with sliding windows and spectral masks. In *Twelfth Annual Conference of the International Speech Communication Association*.

Grais, E. M. & Erdoğan, H. (2012). Gaussian mixture gain priors for regularized nonnegative matrix factorization in single-channel source separation. ISCA.

Grais, E. M. & Erdogan, H. (2012a). Hidden markov models as priors for regularized nonnegative matrix factorization in single-channel source separation. In *Thirteenth Annual Conference of the International Speech Communication Association*.

Grais, E. M. & Erdogan, H. (2012b). Spectro-temporal post-smoothing in nmf based single-channel source separation. In *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, (pp. 584–588). IEEE.

Grais, E. M. & Erdogan, H. (2013). Regularized nonnegative matrix factorization using gaussian mixture priors for supervised single channel source separation. *Computer Speech & Language*, *27*(3), 746–762.

Grais, E. M. & Erdoğan, H. (2013). Spectro-temporal post-enhancement using mmse estimation in nmf based single-channel source separation. In *Interspeech*. ISCA (International Speech Communication Association).

Grais, E. M., Sen, M. U., & Erdogan, H. (2014). Deep neural networks for single channel source separation. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (pp. 3734–3738).

Granhag, P. A. & Hartwig, M. (2008). A new theoretical perspective on deception detection: On the psychology of instrumental mind-reading. *Psychology, Crime & Law*, *14*(3), 189–200.

Gross, S. & Warden, R. (2012). Exonerations in the united states, 1989 - 2012. Technical report, National Registry of Exonerations.

Guadagno, R., Okdie, B., & Kruse, S. (2012). Dating deception: Gender, online dating, and exaggerated self-presentation. *Comput. Hum. Behav.*, *28*(2), 642–647.

Guo, J., Fan, Y., Ji, X., & Cheng, X. (2019). Matchzoo: A learning, practicing, and developing system for neural text matching. In *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, (pp. 1297–1300)., New York, NY, USA. ACM.

Hauch, V., Blandón-Gitlin, I., Masip, J., & Sporer, S. L. (2015). Are computers effective lie detectors? a meta-analysis of linguistic cues to deception. *Personality and social psychology Review*, *19*(4), 307–342.

Hershey, J. R., Rennie, S. J., Olsen, P. A., & Kristjansson, T. T. (2010). Superhuman multi-talker speech recognition: A graphical modeling approach. *Computer Speech & Language*, *24*(1), 45–66.

Hillman, J., Vrij, A., & Mann, S. (2012). Um ... they were wearing ...: The effect of deception on specific hand gestures. *Legal and Criminological Psychology*, *17*(2), 336–345.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, *29*(6), 82–97.

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, *18*(7), 1527–1554.

Hirschberg, J., Benus, S., Brenier, J., Enos, F., Friedman, S., Gilman, S., Gir, C., Graciarena, G., Kathol, A., & Michaelis, L. (2005). Distinguishing deceptive from non-deceptive speech. In *In Proceedings of Interspeech 2005 - Eurospeech*, (pp. 1833–1836).

Ho, S. & Hollister, J. M. (2013). Guess who? an empirical study of gender decep-

tion and detection in computer-mediated communication. *Proceedings of the American Society for Information Science and Technology*, *50*(1), 1–4.

Hoffman, M., Bach, F. R., & Blei, D. M. (2010). Online learning for latent dirichlet allocation. In *advances in neural information processing systems*, (pp. 856–864).

Hu, B., Lu, Z., Li, H., & Chen, Q. (2014). Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*, (pp. 2042–2050).

Jaiswal, M., Tabibu, S., & Bajpai, R. (2016). The truth and nothing but the truth: Multimodal analysis for deception detection. In *Proceedings of the 2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*.

Jensen, M., Meservy, T., Burgoon, J., & Nunamaker, J. (2010). Automatic, multimodal evaluation of human interaction. *Group Decision and Negotiation*, *19*(4), 367–389.

Jindal, R., Malhotra, R., & Jain, A. (2015). Techniques for text classification: Literature review and current trends. *webology*, *12*(2), 1.

Joinson, A. N. & Dietz-Uhler, B. (2002). Explanations for the perpetration of and reactions to deception in a virtual community. *Social Science Computer Review*, *20*(3), 275–289.

Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). Bag of tricks for efficient text classification.

Karimi, H., Tang, J., & Li, Y. (2018). Toward end-to-end deception detection in videos. In *2018 IEEE International Conference on Big Data (Big Data)*, (pp. 1278–1283). IEEE.

Kawahara, H., Takahashi, T., Morise, M., & Banno, H. (2009). Development of exploratory research tools based on tandem-straight. In *Proceedings: APSIPA ASC 2009: Asia-Pacific Signal and Information Processing Association, 2009 Annual Summit and Conference*, (pp. 111–120). Asia-Pacific Signal and Information Processing Association, 2009 Annual Summit and Conference, International Organizing Committee.

Keerthi, S. S. & DeCoste, D. (2005). A modified finite newton method for fast solution of large scale linear svms. *Journal of Machine Learning Research*, *6*(Mar), 341–361.

Kenter, T. & De Rijke, M. (2015). Short text similarity with word embeddings. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, (pp. 1411–1420). ACM.

Kilic, E., Ates, N., Karakaya, A., & Sahin, D. O. (2015). Two new feature extraction methods for text classification: Tesdf and sadf. In *Signal Processing and Communications Applications Conference (SIU), 2015 23th*, (pp. 475–478). IEEE.

Kılınç, D., Özçift, A., Bozyigit, F., Yıldırım, P., Yücalar, F., & Borandag, E. (2017). Ttc-3600: A new benchmark dataset for turkish text categorization. *Journal of Information Science*, *43*(2), 174–185.

Koo, T., Carreras, X., & Collins, M. (2008). Simple semi-supervised dependency parsing. In *In Proc. ACL/HLT*.

Krishnamurthy, G., Majumder, N., Poria, S., & Cambria, E. (2018). A deep learning approach for multimodal deception detection. *arXiv preprint arXiv:1803.00344*, *2018*.

Kristjansson, T., Attias, H., & Hershey, J. (2004). Single microphone source separation using high resolution signal reconstruction. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, (pp. ii–817). IEEE.

Kusner, M., Sun, Y., Kolkin, N., & Weinberger, K. (2015). From word embeddings to document distances. In *International Conference on Machine Learning*, (pp. 957–966).

Larcker, D. F. & Zakolyukina, A. A. (2012). Detecting deceptive discussions in conference calls. *Journal of Accounting Research*, *50*(2), 495–540.

Le, Q. & Mikolov, T. (2014). Distributed representations of sentences and documents. In *International Conference on Machine Learning*, (pp. 1188–1196).

Lee, D. D. & Seung, H. S. (2001). Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, (pp. 556–562).

Li, J., Ott, M., Cardie, C., & Hovy, E. (2014). Towards a general rule for identifying deceptive opinion spam. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, Maryland.

Liu, B., Zhang, T., Niu, D., Lin, J., Lai, K., & Xu, Y. (2018). Matching long text documents via graph convolutional networks.

Lu, S., Tsechpenakis, G., Metaxas, D., Jensen, M., & Kruse, J. (2005). Blob analysis of the head and hands: A method for deception detection. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS'05)*, HICSS '05, (pp. 20–29)., Washington, DC, USA. IEEE Computer Society.

Maricchiolo, F., Gnisci, A., & Bonaiuto, M. (2012). Coding hand gestures: A reliable taxonomy and a multi-media support. In A. Esposito, A. Esposito, A. Vinciarelli, R. Hoffmann, & V. Muller (Eds.), *Cognitive Behavioural Systems*, volume 7403 of *Lecture Notes in Computer Science* (pp. 405–416). Springer Berlin Heidelberg.

MATLAB (2010). *version 7.10.0 (R2010a)*. Natick, Massachusetts: The MathWorks Inc.

McInnes, B. T. & Pedersen, T. (2013). Evaluating measures of semantic similarity and relatedness to disambiguate terms in biomedical text. *Journal of biomedical informatics*, *46*(6), 1116–1124.

Meservy, T., Jensen, M., Kruse, J., Twitchell, D., Tsechpenakis, G., Burgoon, J., Metaxas, D., & Nunamaker, J. (2005). Deception detection through automatic, unobtrusive analysis of nonverbal behavior. *IEEE Intelligent Systems*, *20*(5), 36–43.

Mihalcea, R. & Burzo, M. (2012). Towards multimodal deception detection – step 1: Building a collection of deceptive videos. In *Proceedings of the 14th ACM International Conference on Multimodal Interaction*, ICMI '12, (pp. 189–192)., New York, NY, USA. ACM.

Mihalcea, R. & Strapparava, C. (2009). The lie detector: Explorations in the automatic recognition of deceptive language. In *Proceedings of the Association for Computational Linguistics (ACL 2009)*, Singapore.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013a). Efficient estimation of word representations in vector space. *CoRR*, *abs/1301.3781*.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013b). Efficient estimation of word representations in vector space.

Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., & Khudanpur, S. (2010). Re-

current neural network based language model. In Kobayashi, T., Hirose, K., & Nakamura, S. (Eds.), *INTERSPEECH*, (pp. 1045–1048). ISCA.

Mikolov, T., Kombrink, S., Burget, L., Cernocký, J., & Khudanpur, S. (2011). Extensions of recurrent neural network language model. In *ICASSP*, (pp. 5528–5531). IEEE.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013a). Distributed representations of words and phrases and their compositionality. In Burges, C. J. C., Bottou, L., Ghahramani, Z., & Weinberger, K. Q. (Eds.), *NIPS*, (pp. 3111–3119).

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, (pp. 3111–3119).

Mikolov, T., tau Yih, W., & Zweig, G. (2013). Linguistic regularities in continuous space word representations. In *HLT-NAACL*, (pp. 746–751). The Association for Computational Linguistics.

Mnih, A. & Hinton, G. (2008). A Scalable Hierarchical Distributed Language Model. In *Advances in Neural Information Processing Systems*, volume 21.

Morin, F. & Bengio, Y. (2005). Hierarchical probabilistic neural network language model. In *AISTATS'05*, (pp. 246–252).

Newman, M., Pennebaker, J., Berry, D., & Richards, J. (2003). Lying words: Predicting deception from linguistic styles. *Personality and Social Psychology Bulletin*, *29*.

Nunamaker, J., Burgoon, J., Twyman, N., Proudfoot, J., Schuetzler, R., & Giboney, J. (2012). Establishing a foundation for automated human credibility screening. In *2012 IEEE International Conference on Intelligence and Security Informatics (ISI)*, (pp. 202–211).

Ott, M., Choi, Y., Cardie, C., & Hancock, J. (2011). Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, (pp. 309–319)., Stroudsburg, PA, USA. Association for Computational Linguistics.

Owayjan, M., Kashour, A., AlHaddad, N., Fadel, M., & AlSouki, G. (2012). The design and development of a lie detection system using facial micro-expressions. In *2012 2nd International Conference on Advances in Computational Tools for Engineering Applications (ACTEA)*, (pp. 33–38).

Ozerov, A., Févotte, C., & Charbit, M. (2009). Factorial scaled hidden markov model for polyphonic audio representation and source separation. In *2009 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, (pp. 121–124). IEEE.

Pandit, S., Gupta, S., et al. (2011). A comparative study on distance measuring approaches for clustering. *International Journal of Research in Computer Science*, *2*(1), 29–31.

Pang, L., Lan, Y., Guo, J., Xu, J., Wan, S., & Cheng, X. (2016). Text matching as image recognition. In *AAAI*, (pp. 2793–2799).

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., & Lerer, A. (2017). Automatic differentiation in PyTorch.

Paul, E. (2003). Darwin, deception, and facial expression. *Annals of the New*

York Academy of Sciences, *1000*(EMOTIONS INSIDE OUT: 130 Years after Darwin's The Expression of the Emotions in Man and Animals), 205–221.

Pavlidis, I., Eberhardt, N., & Levine, J. (2002). Human behaviour: Seeing through the face of deception. *Nature*, *415*(6867).

Pedersen, T., Pakhomov, S. V., Patwardhan, S., & Chute, C. G. (2007). Measures of semantic similarity and relatedness in the biomedical domain. *Journal of biomedical informatics*, *40*(3), 288–299.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Pennebaker, J. & Francis, M. (1999). Linguistic inquiry and word count: LIWC. Erlbaum Publishers.

Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (pp. 1532–1543).

Pérez-Rosas, V., Abouelenien, M., Mihalcea, R., & Burzo, M. (2015). Deception detection using real-life trial data. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, (pp. 59–66). ACM.

Pérez-Rosas, V., Mihalcea, R., Narvaez, A., & Burzo, M. (2014). A multimodal dataset for deception detection. In *Proceedings of the Conference on Language Resources and Evaluations (LREC 2014)*, Reykjavik, Iceland.

Pfister, T. & Pietikäinen, M. (2012). Electronic imaging & signal processing automatic identification of facial clues to lies. *SPIE Newsroom*, *2012*.

Radfar, M. & Dansereau, R. (2007). Long-term gain estimation in model-based single channel speech separation. In *2007 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, (pp. 143–146). IEEE.

Radfar, M., Wong, W., Chan, W.-Y., & Dansereau, R. (2009). Gain estimation in model-based single channel speech separation. In *2009 IEEE International Workshop on Machine Learning for Signal Processing*, (pp. 1–5). IEEE.

Radfar, M. H., Dansereau, R. M., & Chan, W.-Y. (2010). Monaural speech separation based on gain adapted minimum mean square error estimation. *Journal of Signal Processing Systems*, *61*(1), 21–37.

Radfar, M. H., Wong, W., Dansereau, R. M., & Chan, W.-Y. (2010). Scaled factorial hidden markov models: A new technique for compensating gain differences in model-based single channel speech separation. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, (pp. 1918–1921). IEEE.

Ratinov, L. & Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, CoNLL '09, (pp. 147–155)., Stroudsburg, PA, USA. Association for Computational Linguistics.

Reddy, A. M. & Raj, B. (2004). A minimum mean squared error estimator for single channel speaker separation. In *Eighth International Conference on Spoken Language Processing*.

Reddy, A. M. & Raj, B. (2007). Soft mask methods for single-channel speaker separation. *IEEE Transactions on Audio, Speech, and Language Processing*,

15(6), 1766–1776.

Řehůřek, R. & Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, (pp. 45–50)., Valletta, Malta. ELRA. `http://is.muni.cz/publication/884893/en`.

Rifkin, R. & Klautau, A. (2004). In defense of one-vs-all classification. *Journal of machine learning research*, *5*(Jan), 101–141.

Robertson, S., Zaragoza, H., et al. (2009). The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, *3*(4), 333–389.

Roweis, S. T. (2001). One microphone source separation. In *Advances in neural information processing systems*, (pp. 793–799).

Sahin, G. (2017). Turkish document classification based on word2vec and svm classifier. In *Signal Processing and Communications Applications Conference (SIU), 2017 25th*, (pp. 1–4). IEEE.

Sahin, H. B., Tirkaz, C., Yildiz, E., Eren, M. T., & Sonmez, O. (2017). Automatically annotated turkish corpus for named entity recognition and text categorization using large-scale gazetteers.

Sak, H., Güngör, T., & Saraçlar, M. (2008). Turkish language resources: Morphological parser, morphological disambiguator and web corpus. In *GoTAL 2008*, volume 5221 of *LNCS*, (pp. 417–427). Springer.

Schmidt, M. (2012 (accessed 2014)). minfunc library.

Schmidt, M. N. & Olsson, R. K. (2006). Single-channel speech separation using sparse non-negative matrix factorization. In *Ninth International Conference on Spoken Language Processing*.

Sen, M., Perez-Rosas, V., Yanikoglu, B., Abouelenien, M., Burzo, M., & Mihalcea, R. (2020). Multimodal deception detection using real-life trial data. *IEEE Transactions on Affective Computing*, (01), 1–1.

Sen, M. U., Erdinc, H. Y., Yavuzalp, B., & Ganiz, M. C. (2019). Combining lexical and semantic similarity methods for news article matching. In *Data Science–Analytics and Applications* (pp. 29–35). Springer.

Sen, M. U. & Erdogan, H. (2014). Learning word representations for turkish. In *2014 22nd Signal Processing and Communications Applications Conference (SIU)*, (pp. 1742–1745).

Sen, M. U. & Yanıkoğlu, B. (2018). Document classification of suder turkish news corpora. In *2018 26th Signal Processing and Communications Applications Conference (SIU)*, (pp. 1–4).

Society, P. (2009 (accessed July 25, 2020)). Piano society.

software, G. I.-T. S. (2009 (accessed 2014)). G.191 itu-t stl software.

Streeter, L. A., Krauss, R. M., Geller, V., Olson, C., & Apple, W. (1977). Pitch changes during attempted deception. *Journal of personality and social psychology*, *35*(5), 345.

Su, L. & Levine, M. (2016). Does "lie to me" lie to you? an evaluation of facial clues to high-stakes deception. *Computer Vision and Image Understanding*, *147*, 52–68.

Sumriddetchkajorn, S. & Somboonkaew, A. (2011). Thermal analyzer enables improved lie detection in criminal-suspect interrogations. In *SPIE Newsroom: Defense & Security*.

Tan, Z.-H. & Lindberg, B. (2010). Low-complexity variable frame rate analysis for speech recognition and voice activity detection. *IEEE Journal of Selected Topics in Signal Processing*, *4*(5), 798–807.

ten Brinke, L. & Porter, S. (2012). Cry me a river: Identifying the behavioral consequences of extremely high-stakes interpersonal deception. *Law and Human Behavior*, *36*(6), 469.

ten Brinke, L., Stimson, D., & Carney, D. R. (2014). Some evidence for unconscious lie detection. *Psychological Science*, *2014*, 0956797614524421.

Tian, Y., Kanade, T., & Cohn, J. (2005). Facial expression analysis. In *Handbook of Face Recognition* (pp. 247–275). Springer New York.

Toma, C. & Hancock, J. (2010). Reading between the lines: linguistic cues to deception in online dating profiles. In *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work*, CSCW '10, (pp. 5–8)., New York, NY, USA. ACM.

Tsechpenakis, G., Metaxas, D., Adkins, M., Kruse, J., Burgoon, J., Jensen, M., Meservy, T., Twitchell, D., Deokar, A., & Nunamaker, J. (2005). Hmm-based deception recognition from visual cues. In *IEEE International Conference on Multimedia and Expo, 2005. ICME 2005*, (pp. 824–827).

Tüfekci, P., Uzun, E., & Sevinç, B. (2012). Text classification of web based news articles by using turkish grammatical features. In *Signal Processing and Communications Applications Conference (SIU), 2012 20th*, (pp. 1–4). IEEE.

Turc, I., Chang, M.-W., Lee, K., & Toutanova, K. (2019). Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962v2*, *2019*.

Twyman, N. W., Elkins, A., & Burgoon, J. K. (2011). A rigidity detection system for the guilty knowledge test. In *HICSS-44 Symposium on Credibility Assessment and Information Quality in Government and Business*. Citeseer.

Venkatesh, S., Ramachandra, R., & Bours, P. (2019). Robust algorithm for multimodal deception detection. In *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, (pp. 534–537). IEEE.

Vincent, E., Gribonval, R., & Févotte, C. (2006). Performance measurement in blind audio source separation. *IEEE transactions on audio, speech, and language processing*, *14*(4), 1462–1469.

Virtanen, T. (2006). Speech recognition using factorial hidden markov models for separation in the feature space. In *Ninth International Conference on Spoken Language Processing*.

Virtanen, T. (2007). Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria. *IEEE transactions on audio, speech, and language processing*, *15*(3), 1066–1074.

Vrij, A. (2001). *Detecting Lies and Deceit: The Psychology of Lying and the Implications for Professional Practice*. Wiley series in the psychology of crime, policing and law. Wiley.

Vrij, A. & Mann, S. (2001). Telling and detecting lies in a high-stake situation: the case of a convicted murderer. *Applied Cognitive Psychology: The Official Journal of the Society for Applied Research in Memory and Cognition*, *15*(2), 187–203.

Wang, Y. & Wang, D. (2012). Cocktail party processing via structured prediction. In *Advances in Neural Information Processing Systems*, (pp. 224–232).

Warkentin, D., Woodworth, M., Hancock, J., & Cormier, N. (2010). Warrants and deception in computer mediated communication. In *Proceedings of the 2010 ACM Cnference on Computer Supported Cooperative Work*, (pp. 9–12). ACM.

wikipedia (2013). trwikimedia dump progress on 20131221. `http://dumps.wikimedia.org/trwikimedia/20131221/`. Accessed: 2013-12-25.

Wu, Z., Singh, B., Davis, L. S., & Subrahmanian, V. (2017). Deception detection in videos. *arXiv preprint arXiv:1712.04415*, *2017*.

Xu, Q. & Zhao, H. (2012). Using deep linguistic features for finding deceptive opinion spam. In *Proceedings of COLING 2012: Posters*, (pp. 1341–1350)., Mumbai, India. The COLING 2012 Organizing Committee.

Yancheva, M. & Rudzicz, F. (2013). Automatic detection of deception in child-produced speech using syntactic complexity features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (pp. 944–953)., Sofia, Bulgaria. Association for Computational Linguistics.

Yin, W., Schütze, H., Xiang, B., & Zhou, B. (2016). Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics*, *4*, 259–272.

Zhou, L., Burgoon, J. K., Nunamaker, J. F., & Twitchell, D. (2004). Automating linguistics-based cues for detecting deception in text-based asynchronous computer-mediated communications. *Group decision and negotiation*, *13*(1), 81–106.

Zhou, L., Burgoon, J. K., Twitchell, D. P., Qin, T., & Nunamaker Jr, J. F. (2004). A comparison of classification methods for predicting deception in computer-mediated communication. *Journal of Management Information Systems*, *20*(4), 139–166.