Detecting Deception using Natural Language

Mattias Appelgren

MInf Project (Part 1) Report

Master of Informatics
School of Informatics
University of Edinburgh

2016

Abstract

A review of the area of machine learning approaches to deception detection is performed. Key datasets and approaches are laid out focusing mainly on the use of speech and text to detect deception. New experiments are performed on the Columbia-SRI-Colorado (CSC) corpus of deceptive speech. These look at features extracted from the transcription. A Random Forest classifier using 27 n-gram features classified at 65.9% accuracy, a vast improvement over previous results with comparable features. For further work a neural network using word embeddings for the transcription is proposed together with ways of incorporating acoustic and prosodic features and using speaker adaptation to improve results.

Acknowledgements

Thank you to Steve Renals and Cathrine Lai for guiding me through this ordeal. I could not have wished for better supervision.

Thank you to Matus Falis for many informative discussions and great cups of tea.

And finally, thank you to Connie Crowe for proof reading and providing helpful comments.

This work is dedicated to my Grandfather, Norman Turner, one of the smartest men I ever met, I cannot remember a time when he did not have a book close by. I will always remember his intellectual sense of humour, even if the punchlines based on Latin needed to be explained to me. You will be sorely missed.

Table of Contents

Chapter 1

Introduction

The detection of deception has long been of interest to psychologists and law enforcement who have wished to detect lies out of academic curiosity and as a professional requirement. In recent times deception detection has had a resurge in popularity with one of the subchallenges in the 2016 Interspeech challenge being deception, as well as a general increase in awareness of security and a fear of the dark. Psychologists have proposed a number of cues to deception, generally resulting from the cognitive and emotional load that they say lying causes [9]. Using technology to detect deception has also been of interest. The most famous example is the polygraph. It measures physical responses when the subject answers questions, which are meant to indicate deception. The machine is controversial as the results are far from perfect and known methods can be employed to cheat the test [52].

Not only are polygraph tests bad, they are also highly intrusive. Equipment must be strapped onto the subject and specific control questions need to be asked. For this reason approaches that use different forms of data are of interest, for example text, speech, or video. Many of the cues found by psychologists are facial expressions, or micro expressions, and body language so machine vision can be used to detect such cues on video. It is also thought that the way we speak changes when we lie. High pitch and emotion may be indications of deception [9]. These can be found in the speech signal. A study found also that we tend to use different words when lying [9].

Performing machine learning on deception is difficult. The first problem is that finding data that is tagged for deceptiveness is hard. People are bad at detecting deception, so the gold standard will never be a set of labels given by human judges as it would be in part of speech tagging for example. Because of this, the size of datasets is often small, with 3 or 7 hours of subject speech. The problem itself is highly subject-dependent with cues to deception changing from person to person or from subject area to subject area. For example, spontaneous lying may cause people to speak slower due to the cognitive load while practised lies may allow you to speak faster.

This project will focus on using speech data to classify deception. I used a dataset of deceptive speech recorded by Hirschberg et al. [21]. It contains 32 interviews where each person was given a monetary incentive to lie in some parts of the interview. The

data consists of two channels of speech, one for the interviewer and one for the subject each of which is transcribed, and tagged for truthfulness at the utterance level. The best published result on the dataset is 66.4% using subject-dependent features and 64.0% using only subject independent features over a baseline of 60.2%.

As this is a two year project, in the first year I focus on the transcription and lexical features. In the past most work has focused on acoustic and prosodic features, the lexical features that were used were taken from the Linguistic Inquiry and Word Count (LIWC) program [41] and a sentiment analysis program. No work was done using other lexical features on the CSC corpus. I have looked at n-gram, part of speech, sentiment, and other lexical features.

This dissertation begins with a literature review of the area of deception detection. I cover cues to deception described by psychologists, datasets used for deception detection, and different machine learning approaches that have been applied to the problem. In Chapter 3 I describe the CSC corpus and the feature extraction that has been done and the classifiers used. Naive Bayes and Random Forests are the classifiers that were tried. Chapter 4 describes the experiments that have been run. The Naive Bayes classifier was used to try the potency of each feature, with poor results. Better results were gained using feature selection. Features were selected using "F-value" and χ^2 and a result of 62.5% accuracy was given by using only 4 features. More feature selection gave 27 features that a Random Forest used to classify at 65.9% accuracy, an improvement 5 times higher than the previous best improvement from lexical features and better than the best published result for subject independent classifiers on the CSC corpus. Additional experiments were done to explore the impact of the prior in Naive Bayes classifiers and by training separate classifiers on each subject to create a subject independent classifier. These results are further discussed in Chapter 5 while chapter 6 proposes future work to be done in the second half of this project.

Chapter 2

Background

Deception is an involved social process. To successfully deceive, you need to be able to invent a story that is internally consistent and plausible, and you need to hide your true intentions or emotions [9]. This leads to a difficult process where your whole social arsenal may be needed to succeed. While deceiving is a complicated task, so is detecting deception. Human judges perform extremely badly on the task, often below chance level [14]. When we try to detect deception we are making the assumption that deceptive behaviour is somehow different than truthful behaviour. Psychological research seems to support this claim as many cues to deception have been found [9]. These stem from the emotions and cognitive difficulty associated with deception (see section 2.2). Paralinguistics and social signal processing seek to understand and detect these type of phenomena and deception falls under the same umbrella [44].

2.1 Paralinguistics

Linguistics and computational approaches to linguistic problems, such as natural language processing (NLP) and speech processing, concern themselves with the syntax and semantics of words, texts, and speech but leave out how things are said, which is an important part of communication. People can draw many conclusions from how things are said that cannot easily be extracted from only the words [44]. For example, people quickly jump to conclusions about personalities and intentions based on very little information. As little as 100ms can be enough for an assessment to be made based on body language as well as voice qualities [36]. In fact, prosodic information can be enough to predict human judgements of character [36]. This is why paralinguistics and social signal processing is important, people obtain much information from non-verbal cues. If we focus only on syntax and semantics, all that information will be lost.

Two types of attributes can influence how we speak: long term traits and short term states [44]. Short term states are factors that can change quickly and have to do with the state of mind of the speaker, who the speaker is addressing, or where the speaker is speaking. This includes emotions and emotion related phenomena such as stress, uncertainty, sarcasm, and deception [44]. Long-term traits are more persistent. They

may change over time but will be consistent in the short term. These include biological traits such as height, age, and gender, group memberships such as culture or social class, and general personality traits [44]. Some factors fall in between the two classes as they last longer than states but not as long as traits. For example, tiredness, intoxication, illness or the social status in a group are not clearly in either category.

Studying these attributes can allow us to improve human-computer interactions in many ways. Where a person might detect anger in a customer over the phone, a standard voice portal does not. Realising the anger of the customer could allow the person, or system, to modify their dialogue in an appropriate way or send the person on to someone who is better qualified to deal with the situation, such as a manager [44]. Different adaptations of the system could be made depending on other metrics such as the age of the caller, which may change the dialogue, or gender, which could allow the system to use a better automatic speech recognition model [44]. Surveillance or various screening procedures are also possible applications [44].

To further work in any machine learning area, having consistent measures, data sets, and units is very important. In paralinguistics this has largely been supplied by the Interspeech challenges [46]. Through the challenge a large number of different tasks have been tackled including emotion [46, 47, 49], personality [47], conflict [49], cognitive load [48], and, in the latest challenge, deception detection [45]. At the time of writing the deception challenge has not been completed. We will look at approaches to two related tasks, cognitive load and emotion detection.

The general approach to emotion detection in speech starts by extracting low level features from the speech signal in small windowed segments [44]. These features, such as energy, pitch, and Mel Frequency Cepstral Coefficients, can either be used as they are or can be further processed to create higher level features. These are often on a higher level, such as the whole segment of interest, and include statistical functions of the lower level features such as maximums/minimums, mean, standard deviation, peaks, and many more [44]. These are included in the openSMILE [16] feature extraction software. The features are then used in machine learning algorithms for classification or regression. Many features are often filtered away or ignored, the best features may be picked by feature selection algorithms [44]. Common approaches include Support Vector Machines, simple classifiers like Naive Bayes, and Hidden Markov Models [44]. Recently Neural Networks have been applied to the problem, often using Long Short-Term Memory (LSTM) networks [56]. They have been useful in approaches that incorporate both a wide context and several types of information, such as audio and video [56].

Speech is a cognitively demanding task that requires the use of our short term memory. If we perform other tasks that require the same resources, our ability to speak coherently can be impacted [23]. Three parts of our speech may be affected: our ability to plan our sentences, our ability to monitor what we are saying, and it may directly impact the muscles used for speaking [23]. These effects could manifest themselves as slower speaking rate or fluency, less clear articulation, or changes to the spectral character of the speech [23]. Approaches to cognitive load classification are similar to those for the emotion task. Features are those extracted by the openSMILE program,

or similar software [26, 37, 50, 23]. Good results have been found using ensembles of ensembles of classifiers including SVMs, Random Forests (RF), and Deep Neural Networks (DNN) [26], as well as algorithms such as AdaBoost [18]. Speaker-dependent systems were found to be important as the way cognitive load affects people differs greatly [23].

2.2 Cues to deception

Lying can take many forms, from white lies to attempts to avoid punishment for serious crimes. While the majority of lies being told are harmless and are simply designed to gain some small social benefit like saving face, through laughing when someone tells a hurtful joke or giving a false compliment to a friend [9], others can be more harmful. People deceive their way out of punishment or into places to which they should not have access [34]. The consequence of this kind of deception can be quite profound, for example, a social engineer may gain access to sensitive data through careful use of deception. It is a fact that these things work and that people can be manipulated, the human factor is often quoted to be the biggest security flaw, much harder to deal with than your average computer bug [35]. For reasons such as these detecting deception is important.

So how do we go about detecting deception? Quite a bit of research has gone into finding cues to deception, this is based on the idea that people act differently when they are lying. If we can find these differences then maybe we can work out when someone is lying. The polygraph, for example, is based on the idea that lying will cause a physical response that can be detected to indicate untruths. Whether or not polygraphs work is debatable and some even call it pseudoscience [25]. There is little doubt that polygraphs are not perfect, as you can never be 100% sure about the results, and even then there are many known ways to cheat them [52]. The other problem that a polygraph faces is its intrusiveness. Having to wire people up to a machine would not work if you needed to screen people at a job interview or over the telephone.

The idea that people act differently when they lie is a good starting point for thinking about deception detection. So how does this manifest itself, what kind of differences can we expect? Again, this brings us back to the idea of cues. These take different forms depending on what part of the lying process they are created by. For example, if part of the lie is hiding what you truly feel about something and you try to hide a facial expression, you might not be able to completely conceal it which may lead to a shorter version or distorted version of the original expression. This is called a leakage cue [12]. It is also proposed that people who are lying will be more aroused or excited [59],or they may feel guilty [59] or proud that they are getting away with their deception [10]. Liars may also be under a bigger cognitive load. They need to keep their stories consistent, perhaps even make it up as they go. This extra load may cause liars to speak slower or, if they have rehearsed their stories, speak faster or with fewer pauses. The stories may also seem less compelling [59, 10].

Deception may also have to do with self-representation [9]. For example, in a scenario

where someone is taking a job interview they may ask questions about a previous employment you were fired from. A natural response could be to lie about what actually happened as you want to present yourself in a good light to your potential employers. Alternately, it could be as easy as sitting up straight and closing the Facebook tab when your boss walks past. In both these situations the deceiver is trying to assume a role with the aim to impress. Playing such a role is completely natural, and most people do, but you can do so while being completely honest at the same time, a parent will act differently in front of their child than with their university friends. On the other hand you can be untruthful with your role, the employee who sat up straight in front of his boss might genuinely care about what the boss thinks of him and go back to work with renewed vigour or he may simply give him a fake smile but then grimace as soon as he turns around and go straight back to his procrastination.

Now, given that a deceiver is deliberately adopting a role that does not reflect his true feelings, he may try to control his actions in a way that seems appropriate for the emotions he is trying to replicate. This deliberateness may be the cue to deception that we are seeking [9]. Through controlling what they say a person may end up sounding insincere, for example, by overcompensate in some manner, in trying to act empathetic they might look unnatural as they put effort into smiling and nodding. “The process may be akin to what happens to experienced typists who try to focus on the location of each of the characters on the keyboard instead of typing in their usual un-self-conscious way.” [9]

The words we say are perhaps one of the most controlled part of our communication, so leakage cues may seem unlikely, however it seems that this is not the case. Although people have control over their words, they can often slip up in subtle or unexpected ways [38]. In a famous news story Susan Smith stated, on television, that her children had been kidnapped saying “My children wanted me. They needed me. And now I can’t help them” [27]. A FBI agent noticed the fact that she was using the past tense, saying “they *needed* me”. Relatives of missing people will generally speak of them in the present, assuming they are still alive. Because of this the agent suspected the mother was already thinking of her children as dead. It turned out that Susan had in fact drowned her children in a lake [38].

It seems that we may be able to extract information from the words people say, but what should we be looking for? Liars may want to distance themselves from their stories. Since they have not actually experienced what they are talking about they might be more reluctant to put themselves directly in the story [38]. This may manifest itself as using less personal pronouns such as “I” or “my” [38]. We have also said that liars may feel guilty about their lies, this may manifest itself as more negative words such as “hate” or “sad” [38]. Finally, since lying could be seen as being cognitively more taxing, because of having to create a coherent story, we might expect liars to use simpler sentences and simple actions over complex justifications or details [38]. An interesting note is that a model of deception in a specific subject area may not generalise well to other subjects [38]. The context where lies appear is very important, so although the underlying reasons might be similar, the way they manifest may be quite different.

Given all of these possible cues there must be a lot for people to go on, so they must be quite good at detecting deception, right? The answer seems to be no [14, 39, 11]. People are surprisingly bad at deception detection, often performing at or close to chance [11]. In a study where people tried classifying lies on the CSC corpus (described below) they found that the mean performance was 58.23%, relatively far below the chance rate of 63.87% [14]. Similar results have been found when it comes to false opinions online [39, 40]. It seems people are biased toward the truth preferring to believe rather than distrust [39]. Now, although this is true, some were able to do much better (and some much worse). The best performance on the CSC corpus was 71.48%, a significant improvement over the baseline [14]. [11] found similar trends where some individuals were able to make much better judgements, generally they had been given some kind of training. For example, criminals and law enforcement perform better than average [1].[1] Since this is the case it seems plausible that we could train a system that would perform lie detection.

2.3 Review of previous work

2.3.1 Data

Work on computational models of deception detection has been limited largely by the available datasets. Where these datasets exist they vary as to quality and size. As with many paralinguistic tasks collecting good data is very difficult. Many deceptive cues from psychology literature arise from a fear of getting caught or a pleasure in successful deception. Setting up experiments with fear as motivation is difficult to justify to an ethics committee, which means other means of motivation must be used [21]. The most common approach is to provide the chance of a prize if the subjects are successful in their deception [21, 13], this may also include a chance to lose out on money if true statements are disbelieved [13]. A different incentive could simply be winning over other participants if the task is a game that is naturally deceptive [5] or the data could even be taken from real world situations such as trials [42]. The way deception occurs can also vary. For example, a role could be assigned to different people either by subject choice [13] or by choice of the researchers or by chance [13, 5]. Alternatively, deception could be about your own performance on a task [21] or each subject performs a task both truthfully and deceptively [33]. The medium can vary from pure text data [33, 39, 30], to transcribed speech [21] or both speech and video [5].

Just as the form of the data varies, so does that goal or task. The variation may stem from the type of data (eg. speech/text) but also in what labels are being classified or what the task was. I categorise the type of goal into two different groups: role classification and lie classification. In role classification the subjects are assigned either a deceptive or non-deceptive role. The deceptive role may be a student who has stolen

[1] Parole officers were found to be the worst at detecting deception scoring 40% with a baseline of 60% in one study [1].

Dataset	Text	Audio	Video	Task	Best result
CSC Deception[21]	T	Y		Truth/Lie	66.4%/64.0%[2] (60.2%)
IDIAP Wolf[5]		Y	Y	Role	0.59 F-score (0.33)
Mihalcea et al. [33]	Y			Truth/Lie	70.8% (50%)
Deceptive review 1 [40]	Y			Truth/Lie	90% (50%)
Deceptive review 2 [30]	Y			Truth/Lie	60-80%[3](50%)
Interspeech DSD[45]		Y		Truth/Lie	70%[4](63%)
Court Cases [42]	T	Y	Y	Truth/Lie	75.2% (50.4%)

Table 2.1: An overview of available deception datasets, what type of data they use, the main task, and the best classification results found on the dataset. The numbers in brackets are the reported baselines. T is for transcription.

an exam key [13], someone at an airport who has a fake bomb in their bag [13], or the werewolf who wants to kill all the villagers in a game of "Are you a Werewolf?" [5]. The task is simply to detect the people with deceptive roles.

In lie classification subjects do not fall into either liars or truthtellers, instead they may perform the same task both truthfully and deceptively [33] or simply have an overall goal which requires both truth and lies [21]. In this type of task the goal is to classify individual answers [21, 33] or interview segments [21] as truths or lies.

The application of the two task types may also be quite different. Deceptive role detection may be used as a screening procedure, for example in border control [13], while lie detection could be better suited for detecting deceptive opinions [39], perhaps in politicians, or as part of an interrogation, perhaps in court [42].

2.3.2 Approaches to Automatic Deception Detection

We can split the features used to detect deception into three main classes: text, speech, and video. I will define text features as anything that can be extracted from text, transcriptions, or output of Automatic Speech Recognition (ASR) systems. Speech features are anything that can be extracted from the speech signal such as prosody or acoustic features, but do not include ASR outputs. The final category includes any body-language features that can only be extracted if you can see the speaker. In this project the main concern is the first two categories but I will mention some results from the third.

Text features are those taken from the words being said. In written text these are the only information available, in speech they could come from a transcription or from the output of an ASR system. We can look at text features in two contexts: part of spoken deception or written deception. Written deception could include text messages, online reviews, or any other written text that may be deceptive. Spoken deception could be, for example, a conversation, an interview, or a speech.

If we look at it as spoken deception, the amount of work done is very limited. Only the CSC corpus [21] (as far as I am aware) contains a transcription of the speech,

making lexical features easier to extract. The main lexical features used there are word categories [21] from the linguistic inquiry and word count LIWC program [41], counting filled pauses [21, 19], sentiment analysis [21], emotional content [15, 19], and syntax related features [19]. Additionally, usage of specific words, such as attempts to be compelling (yes/no), and the presence of qualifiers (really/absolutely) were found to be indicative of deception [15]. Only one paper reports the classification accuracy on only the lexical features: they get classification accuracy 62.0% over a baseline of 61.2%.

With the rise of the internet, publishing writing to a wide audience has become extremely easy. In addition to this the opinions of other people become very important. Online reviews of products and businesses influence the way people make purchases [39]. As their reputation becomes important, companies have incentives to produce deceptive positive reviews for their products [30]. This has spawned work on trying to detect deceptive opinions [40, 30]. [33] has people produce truthful and deceptive opinions on abortion, the death penalty, and their best friends. [40] and [30] have people write deceptive reviews about hotels, restaurants and doctors. On all tasks automatic classifiers are trained that outperform human judges, that perform slightly better than chance. N-gram models were found to outperform LIWC features by a good margin [40]. The words and n-grams used for classification are often highly domain specific, this means that a classifier for deceptive hotel reviews generalises badly to deceptive doctor reviews [30].

Moving on to speech features. For the task of lie detection the CSC corpus [21] contains interviews with long form answers, while Elkins [13] gathered a set of yes/no questions where some questions were meant to evoke emotional deceptive responses. For role classification the IDIAP Wolf corpus [5] contains games of the role playing game “Are you a Werewolf?” and the Deceptive Speech Dataset (DSD) [45] has a set of naughty students. Many different features have been used to classify speech deception. Acoustic features such as Mel-scaled cepstrums were found to make a small improvement [15]. Pitch, energy, and duration are the basic prosodic features used [15, 21] and various statistics have been based off these, such as averages and maximums/minimums over frames or whole utterances [15, 21]. These were also found to help classification, the best result was found when combining acoustic and prosodic features [15].

2.4 Summary

When deliberately deceiving, people change the way they speak and act. Changes stem from trying to both hide and fake emotions as well as the extra cognitive and emotional load required to perform well. Psychological research has focused on finding cues to deception which include changes to voice qualities such as energy and pitch, the speed and coherence of speech, and even word patterns. Machine learning approaches to detect these differences have been applied to a number of different datasets. Some specific domains, such as deceptive opinion detection, have had moderate success but

the problem is far from being solved with the majority of systems performing close to chance.

In the domain of speech transcriptions and deceptive texts LIWC has been frequently used, however, recently unigram and n-gram features have been found to perform as well as LIWC. Sentiment analysis has also been used with some success. Classification has been done with Naive Bayes, Support Vector Machines, and Decision Trees. For speech, different features based on pitch, energy, and duration were used together with acoustic features such as Mel Frequency Cepstral Coefficients. The best results came from combinations of lexical, prosodic, and acoustic features and using additional subject-dependent features (See table 2.1 for details on data sets).

One of the earliest datasets for deception was the CSC corpus. It contains transcribed speech of interviews that is tagged for truthfulness on a low level. The results reported on the dataset were not very impressive, and other datasets have had much better results since then. In this project I will look at some different techniques for detecting deception, taking inspiration from recent work done on deception detection as well as other fields within natural language processing, speech recognition, and paralinguistics.

For the transcription of the CSC corpus, only LIWC and sentiment analysis was really tried since it has been found that n-gram features have had similar or even better results. Furthermore, it was shown that the words and syntax used by deceivers were simpler because of the added cognitive load of lying. I will look at whether or not we can use this information to improve classification. For the speech side, advances have been made in the area of paralinguistics since the CSC corpus was created and work was done on it. The openSMILE feature extraction framework has become very powerful and allows a large number of comprehensive features to be extracted very easily. Finally, subject-dependent features and models have been shown to work very well for deception detection as well as in other areas such as cognitive load detection. The best results on the CSC corpus use subject-dependent features.

This project focuses on the lexical features. The aim is to apply new and different techniques to the CSC corpus in an attempt to improve classification results. My hypothesis is that results as good as those found by Hirschberg et al. can be achieved using simple n-gram features. Additionally, the inclusion of other features based on lexical information may also improve accuracy. To test this I will extract a number of features based on n-grams and the length of sentences and words. Part of speech tags and sentiment will also be tried as features. Part of speech tags may indicate what type of words are being used, separate from what the specific words are. This could be seen as similar to some LIWC categories that look for word types. Sentiment was found to be useful before, so I hope to replicate the results using an open source sentiment analysis tool. Experiments will be run using the Naive Bayes and Random Forest classifiers that have been found to have similar performances to other more complicated classifiers.

Chapter 3

Methodology

3.1 The data

3.1.1 CSC corpus

The CSC corpus [21] consists of 7.2 hours of subject speech, tagged with truthfulness on two levels. It was gathered in an interview scenario where subjects were given a financial incentive to lie in parts of the interview. The data consists of both speech in two channels, interviewer and subject, and a hand transcription that has been time aligned to the speech.

In the study each subject performed six different tasks. They were given a score for each. The scores were compared to a profile based on "the 25 top entrepreneurs of America". The results were manipulated in such a way that the subject score was better than the profile in two tasks, worse in two tasks, and exactly the same in two. They were told that they would get a $100 cash prize if they could convince an interviewer that they had scored exactly as well as the profile. This meant they were motivated to tell the truth in two areas and lie in four. The interviewer had to work out what their actual score was and could ask any question except any of the questions that had been asked during the tasks. The subjects indicated if what they said was true or false by pressing a pedal that was hidden from the interviewer.

Each interview was about 25-50 minutes long and 32 subjects were interviewed. Each subject was a speaker of "Standard American". In total, 15.2 hours of interview was recorded of which 7.2 hours was subject speech. The interview scenario with monetary reward can be seen as a low stakes reason for lying. It may not invoke as many emotional responses as lying from fear or shame would do. However, the lies are spontaneous, rather than planned, and each subject can largely use their own judgement when lying is appropriate for their goal.

The speech is transcribed by human annotators and the transcription is automatically force aligned to the speech using an automatic speech recognition system. The transcription includes additional information above simply words. Sentence-like units are

separated by slashes: '/', sentences that are not finished are marked '/-', words that are interrupted are marked as 't-'. Mispronunciations are tagged with <MP>, laughing is included as <LG>, and breathing as
.

The corpus is tagged in two respects. The "big lie/big truth" tag has to do with the overall truthfulness of a interview segment, for example when the interview is about a task where the subject performed worse than the target profile and she is trying to talk herself up then that whole segment will be tagged as "big lie". All the sentences within these segments may not be lies, however, the aim of the segment is still deception. The second type of tagging is "little lie/little truth". This comes from the data gathered by the pedal. Here individual sentence-like units (SUs) are tagged with truthfulness. For example, if I were a subject and I had said "My name is Bob." that would be tagged as a little lie. The big lie/little lie tagging scheme has nothing to do with how untruthful the specific lie is, it is just the locality of the lie. For this project I have been looking at the little lie tags.

3.2 Preprocessing and cleaning

The first question to answer is how to segment the data. Several ways are possible: each sentence (separated by '/') could be an individual data point, splitting on speaker turns, such that each data point would be an answer to a question, or perhaps going to a lower level and splitting on individual words. The problem with sentence level splitting becomes apparent if we look at an example from the data. Here is an example of a speaker turn that was a lie: "um /
 just guy./ <LG> they kind of broke us down that way./ you have the guys on one side and the girls on the other, and that was about it./". The whole sequence is part of an answer, it is coherent and fits together. If we were to split it apart it would be meaningless. Additionally, the whole sequence is one lie. Each part of the unit may not be untrue, but stringing it all together does make it so. Separation done on speaker turns, rather than sentences or words, was chosen because of these considerations.

The transcription is given in the form of Praat Text Grid files. There is one for truth/lie values and one for each speech channel (left=interviewer, right=subject). The truth/lie text grid has been hand-aligned with the word transcription. The assumption I have made is that the alignment has been done in such a way that any speech that happens during what is tagged as a lie is an individual lie and will thus be its own example in the dataset. Generally this will represent a question and an answer, although only the answer is used. In Figure 3.1 we see what a lie would look like in the text grid. "<SIL>" represents silence and is most often found when the dialogue partner is talking. I extract all the text underneath to create a pair with the raw text and the tag: ("<SIL> outstanding physical prowess. / </BR> I'm just Playing with you. / <LG>", "LIE").

After the raw text had been extracted from the TextGrid, additional cleaning was performed to remove irrelevant information. Off-topic speech is an obvious thing to remove, however, some tags such as <MP> for mispronunciation might be useful to add back in in future work. One significant tag is left in, the <LG> tag which represents

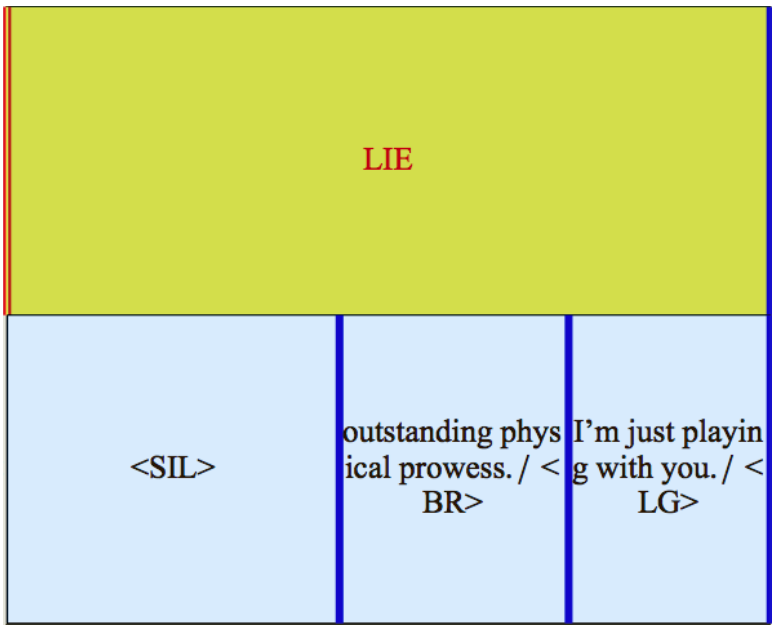

Figure 3.1: A lie in the TextGrid file.

laughter. Here follows a lists of cleaning done to all data:

- Removed big lie information which is included included in the text as "%LU/H%%INTERACTIVE%%84.99621477442952%".
- Removed off-topic speech which was tagged as <OTP> </OTP> (eg. adjusting mic or pedal)
- Removed a number of tags for non-speaking: <SIL>,
, <SN>, <LN>, <ASP>, <MP>, and <∼>.
- Replaced "/-" with SENTENCEDISFLUENCY and any word ending with - with WORDDISFLUENCY.

3.3 Feature Extraction

A number of numeric features are extracted from the raw words, starting with a bag-of-words vector. Recent work has shown that n-gram word counts often work as well or better than the LIWC categories [33, 40, 30]. Next, we extract a bag of part-of-speech (POS) tags. If the word usage changes then this might be detectable in the POS tags. These would also generalise better than counting specific words being said. Sentiment was used by Hirschberg et al. [21]. An analysis program is used to extract sentiment scores on a real valued scale. Word and sentence length statistics are extracted because these may be an indication of the added cognitive load put on the liar. Finally, Hirschberg et al. found filled pauses to be useful for classification, so a total count of filled pauses was extracted.

[21] looked at the number of filled pauses. They found that filled pauses were cor-

related with truth. Thus I counted the total number of filled pauses. The number of "um"s and "uh"s is already counted in the bag-of-words vector, so only total number is included. In [21] they found sentiment to be an important feature for classifying deception. They used Whissell's Dictionary of Affect in Language [55] which gives a numeric value indicating the emotional content of the sentence.

3.3.1 Bag-Of-Words

Two bag-of-words vectors were extracted. The first uses the 1000 most frequent unigrams with stop words removed. The second uses unigram, bigram, and trigram counts and keeps any words with three or more occurences in the dataset. This resulted in a 3306 dimension feature vector. A bag-of-words vector is simply the number of times chosen tokens appear in a given sentence.

3.3.2 Bag of POS tags

A bag-of-POS tags was created by first running each sentence through the basic POS-tagger available in the open source Natural Language Toolkit (NLTK) [3], which uses 32 different POS tags, before using the tags as tokens in a bag-of-words vector. Before tagging, all <LG> tags, disfluency marks, '/', and any symbols that are not full stops were removed.

3.3.3 Sentiment

The vaderSentiment [24] program was used to extract sentiment information about each example. The program gives a score between 0 and 1 in three sentiment categories: positive, negative, and neutral. A combined score that ranges from -1, negative, to +1, positive is also created. Each example was cleaned in the same manner as for POS tagging and then run through the sentiment analyser. This resulted in four real valued features.

3.3.4 Counting words

The average length of words, maximum word length, length of sentence in characters and number of words in a sentences were extracted as features. These may be indications of extra cognitive load leading to shorter sentences or simpler words. There is a small difference in the average length of sentence between lies and truths, 98.5 vs. 102.5, and in number of words, 18.6 vs 19.6.

3.3.5 Filled Pauses

The number of “um”s and “uh”s have been counted separately in the bag-of-words vectors, the combined count is added as an additional feature.

3.4 Classifiers

Two classifiers were used in experiments, the Naive Bayes and Random Forest classifiers. Both were chosen for their simplicity and ability to create complicated non-linear decision surfaces.

3.4.1 Naive Bayes

The Naive Bayes classifier attempts to estimate the conditional probability of a class given data $P(C|\mathbf{x})$. Where C is a random variable representing the classes and x is the input vector or data that the classification is conditioned upon.

Using Bayes' Theorem this can be expanded to:

$$P(C|\mathbf{x}) = \frac{P(\mathbf{x}|C)P(C)}{P(\mathbf{x})}$$

.

Classification is done by picking the class with the highest probability. Since $P(\mathbf{x})$ does not depend on the classes it is often ignored.

$P(C|\mathbf{x})$ is called the posterior probability, $P(C)$ is the prior, and $P(\mathbf{x}|C)$ is the likelihood.

The likelihood multiplied by the prior can be rewritten as the joint probability of the class and the data by applying the product rule:

$$P(\mathbf{x}|C)P(C) = P(\mathbf{x},C) = P(x_1,...,x_n,C)$$

.

Using the chain rule this can be expanded to:

$$\begin{aligned} p(C,x_1,\ldots,x_n) &= p(x_1,\ldots,x_n,C) \\ &= p(x_1|x_2,\ldots,x_n,C)p(x_2,\ldots,x_n,C) \\ &= p(x_1|x_2,\ldots,x_n,C)p(x_2|x_3,\ldots,x_n,C)p(x_3,\ldots,x_n,C) \\ &= \ldots \\ &= p(x_1|x_2,\ldots,x_n,C)p(x_2|x_3,\ldots,x_n,C)\ldots p(x_{n-1}|x_n,C)p(x_n|C)p(C) \end{aligned}$$

All of these conditional probabilities are difficult to estimate which is why the Naive assumption is made. The assumption is that the probability of each dimension x_i is independent of all other dimensions and only depends on the class. This leaves us with the equation:

$$P(C|\mathbf{x}) \propto P(C)P(x_1|C)\dots P(x_n|C) = P(C)\prod_{i=1}^{n} P(x_i|C)$$

This is a strong assumption and has no grounding in observation, it is simply for mathematical convenience and turns out to still work in many applications.

3.4.1.1 Probability Distributions

The individual probabilities $P(x_i|C)$ can be estimated in different ways depending on the type of features being used. If the features are binary then a Bernoulli distribution can be used to estimate the features. Binary features can be seen as representing the presence or absence of a word in a sentence.

If the data is discrete and positive, such as for bag-of-words vectors, a multinomial distribution can be used. This distribution counts the number of times an event occurs in some context, such as the number of times "hello" appears in a sentence.

Finally, for real valued data, the data can either be made binary or discrete placing values in bins. Alternatively a single one-dimensional Gaussian distribution can be trained for each feature.

3.4.1.2 The Prior

The prior models our beliefs about the classes before we observe the data. For example, we may know that there is a 40/60 split between lies and truths which would mean the prior $P(C = Lie) = 0.4$ and $P(C = Truth) = 0.6$. The prior can be learned just like other parameters or set beforehand. The maximum likelihood solution for the prior is estimated simply by $P(C = c) = \frac{n_c}{N}$ where n_c is the number of instances of class c in the training data and N is the total number of training examples.

3.4.2 Random Forest

The random forest classifier [4] is built up by a large number of decision trees that are combined through voting.

3.4.2.1 Decision Trees

The decision tree is a discriminative learning algorithm that is based on the idea of asking simple yes/no questions. The algorithm is set up as a binary tree where each

node is a question and each leaf is a decision. For example, a node could be $x_i > 0$ and a leaf could be the classification label "Truth" or 1. Each instance is classified by going through the decision tree and returning the label given by the leaf node reached.

Building a tree is done by finding the best places to split the dataset in order to separate classes as well as possible. A feature is selected and a question is asked in such a way that the largest amount of separation happens. This is repeated until no more questions can be posed, all leaf nodes contain pure sets, or some other finishing criteria.

Decision trees are powerful as they can completely ignore irrelevant features and outliers in the data. Additionally, they are very transparent as the rules being used can be easily read off, showing which features are important for classification. However, decision trees often generate bad models due to overfitting, which is caused by the way rules are created, since noise can very easily be learned.

3.4.3 Tree aggregation

One way to deal with overfitting is to train many trees on different subsets of the data and pooling their output to create a better prediction. The hope is that, although individual trees may overfit, on average the aggregation of trees will create a better model. Training on different subsets of the data makes sure that the trees are independent and that the same tree is less likely to be trained several times. Classification is simply done by majority voting. One additional way of increasing the randomness and thus the variation in trees is to pick the feature that will be used in a rule at random.

All in all this leads to a simple classifier that can create complex decision boundaries.

Chapter 4

Experiments

Experiments were done using Naive Bayes and Random Forest classifiers. The best results were found using the Random Forest after doing extensive feature selection.

4.1 Naive Bayes

To explore the potency of the features, simple Naive Bayes classifiers are trained on individual features and on all features together. An average of 10-fold cross-validation is reported for each experiment. The baseline is a dummy classifier that classifies everything to "truth", which is the majority class.

In figure 4.1 we see how well our initial Naive Bayes classifier does. Only two of the classifiers actually improve at all over the baseline, the sentence length and number of words features, while most classifiers perform below baseline.

Features	Accuracy %
Baseline	61.78
bag-of-words	58.66
Bag of POS	58.61
Sentiment	60.93
Sentence Length	**61.85**
Number of words	**61.85**
Average Word Length	61.05
Longest Word Length	61.78
Filled Pauses	61.78
All Features	58.78

Table 4.1: Accuracy of the Naive Bayes classifier for different features.

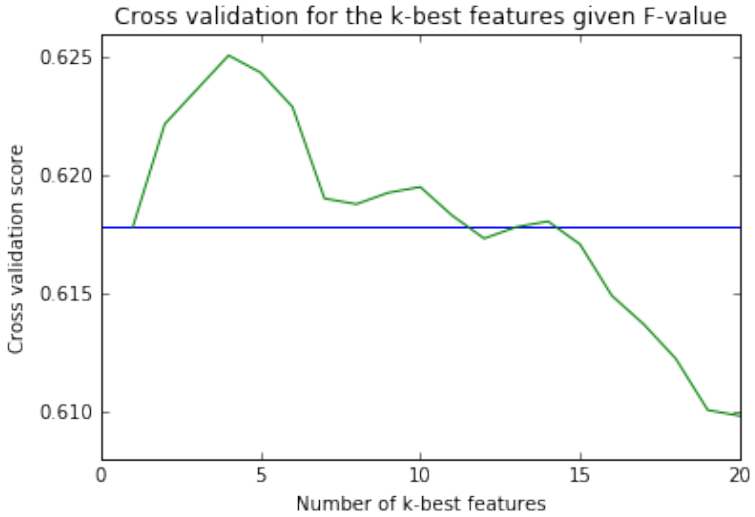

Figure 4.1: Accuracy of the Naive Bayes classifier using the k-best features given f-value. The blue line is the majority class baseline.

4.2 Feature Selection

There are several reasons why feature selection might be a good idea. Firstly, by finding the features that are helpful predictors we find patterns that can indicate in what direction future research could go, for example, through predicting what new features could be extracted. Secondly, if some features are not contributing, the classifiers may be getting bad results as the true indicators of deception are getting lost in the noise. Finally, overfitting is a major problem that needs to be considered. Since many features may have very low counts the probabilities estimated for those features may be very wrong. Selecting only the most important features may improve this result.

In this experiment, features are scored using two metrics, the ”F-value” and χ^2. The k best features are chosen to train a Naive Bayes classifier using 10-fold cross-validation. k is varied from 1 to 20, the accuracy is recorded in figures 4.1 and 4.2 for ”F-value” and χ^2 respectively.

In both experiments we classify over the baseline. Using only four features, an accuracy of 62.5% is achieved, this is an improvement of 0.8% over the baseline. This is as good as the lexical features used by Hirschberg et al. We notice that as we add more features the results tend towards lower accuracy, however, there are several features past the best results that do improve classification accuracy. In the Random Forest feature extraction experiment (section 4.4) we attempt to pick out these features specifically.

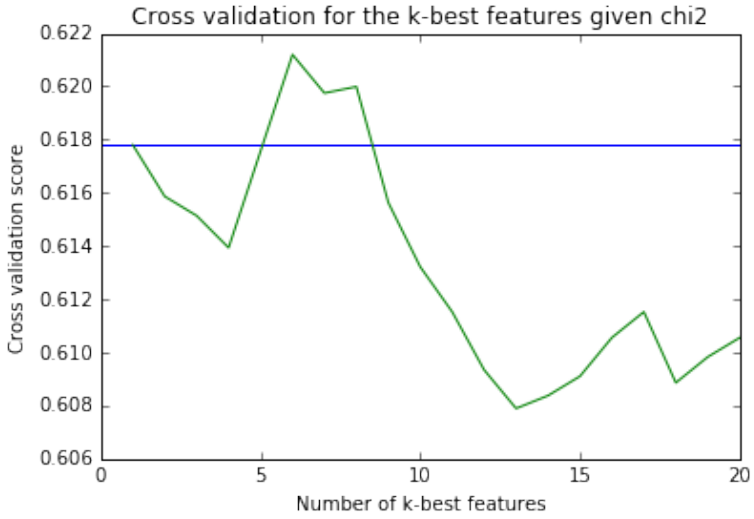

Figure 4.2: Accuracy of the Naive Bayes classifier using the k-best features evaluated with χ^2 (chi2). The blue line is the majority class baseline.

4.3 Shifting Prior

In previous experiments, the prior has been learned by the model. Since the baseline is 61.7% that many examples are truths. This means the prior will be around 0.617 for truths and 0.383 for lies. In this experiment the effect of different prior beliefs is explored. The prior is set to values between 0 and 1 with 0.01 increments. We use the full feature set classifier from the first experiment and the best classifier from the feature selection experiment, the classifier using the four best features given ”F-value”.

Figure 4.3 shows the results for the full feature set. The best accuracy is found at lower lie priors. This is probably because we get closer to doing majority class classification, which is better than classifying below the baseline. Between 0.0 and 0.2 prior the accuracy does go over the baseline, although not to any noteworthy degree. The recall line is more or less diagonal, which indicates that the performance is close to chance level. Precision is never higher than 0.5. Precision does rise when closer to priors favouring truth, most probably because fewer instances are being classified as lies.

Figure 4.4 shows the results for the shifting prior experiment applied to the Naive Bayes classifier using the four best features (see Section 4.2). From the recall curve we can see that when the prior is over 0.5 in favour of lies, the majority of instances are classified as lies and the opposite is true when the prior is under 0.5. This means that the majority of instances are only being classified by the prior. The best precision happens just under 0.4 lie prior. That is about what the learned prior would be as it would be close to 0.383. We see that the approach of this classifier is to simply classify everything to truth unless the few features indicate anything else. The recall gets larger as the prior goes closer to 0.5 lie prior with little impact on accuracy. Precision does falter slightly, so if recall is preferred over precision using a prior slightly higher than 0.4 may pay off.

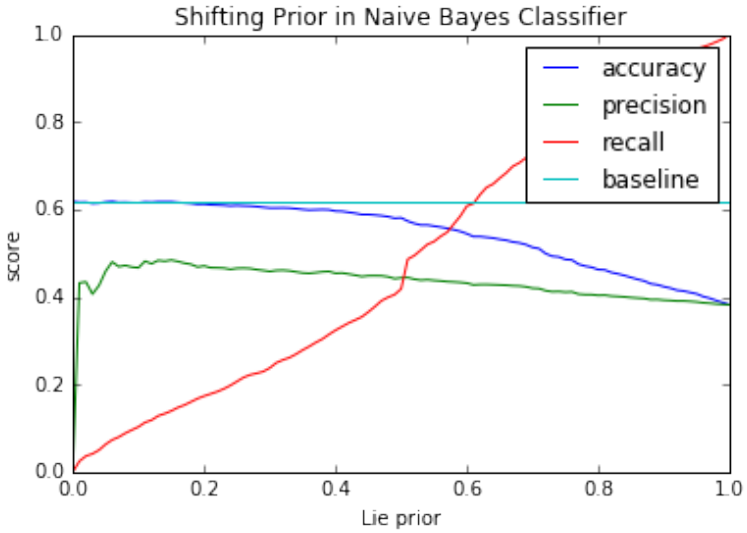

Figure 4.3: Accuracy, precision, and recall of Naive Bayes classifiers initialised with different priors. The baseline shows the accuracy of the majority class classifier.

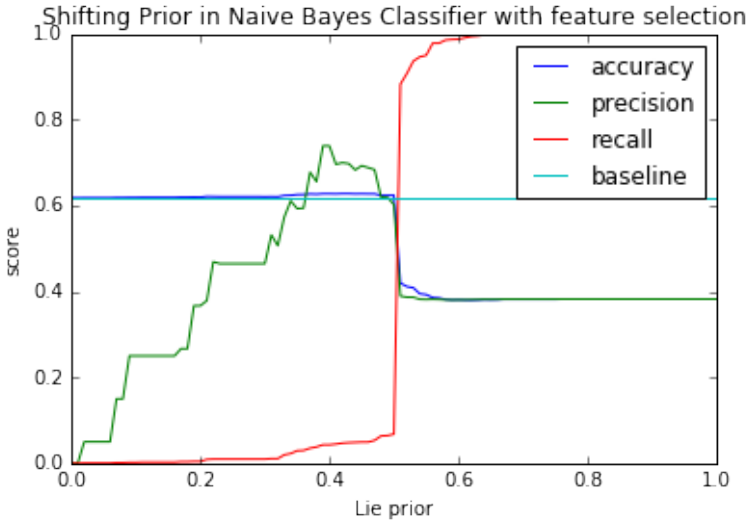

Figure 4.4: Accuracy, precision, and recall for different deception priors using the four best features found by F-value. The baseline is the majority class classifier.

Features	Accuracy
break, poor, mm, did really, violin, just went, think pretty, ninety, exactly, lot friends, good good, lot time, heard, week, don know, gosh, really lg, laugh, um took, kids, know sentencedisfluency like, certain things, saying, case, guess like, like say	65.9%
break, poor, mm, did really, violin, just went, think pretty, ninety, exactly, lot friends, good good, lot time, heard, week, don know, gosh, really lg, laugh, um took, kids, know sentencedisfluency like, certain things, saying, case, guess like, easy	65.8%

Table 4.2: The feature sets granting the best classification accuracy using a Random Forest classifier.

4.4 Feature Selection and Random Forest

If we look at figures 4.1 and 4.2 we see that sometimes adding an extra feature improves the classification result and sometimes the accuracy is reduced. This experiment is created to find the features that improve classification. One way to do this could be to do a search over all feature combinations, however with over 3000 features this is completely infeasible. The approach taken here is this: a classifier with one feature is created for all features. All features that improve classification are kept, the rest are discarded. Next, classifiers using two features are trained. The features are taken from those that were kept in the first step. The feature pairs that improve classification over the previous best score are kept for the next stage. This process is repeated until no more improvements were made. The classifier used in this experiment was a Random Forest. It was found to have better classification accuracy than Naive Bayes for the same features.

After choosing 27 features the experiment terminated. The results can be seen in 4.2. All of the features chosen are simple unigrams, bigrams and trigrams. None of the sentiment features or POS tags are used as features. The accuracy of the best classifier is 65.9%, a vast improvement over the best lexical classifier (see Table 4.4 for details) and better than the best subject-independent results, which uses a combination of lexical, prosodic, and acoustic features.

4.5 Subject-dependency

In previous work the best results were found when using subject-dependent features. The main finding seems to be that people who lie generally just act differently than they do normally. This might, for example, be speaking faster or slower or having higher or lower pitch. Even when using a polygraph the testers first ask some control

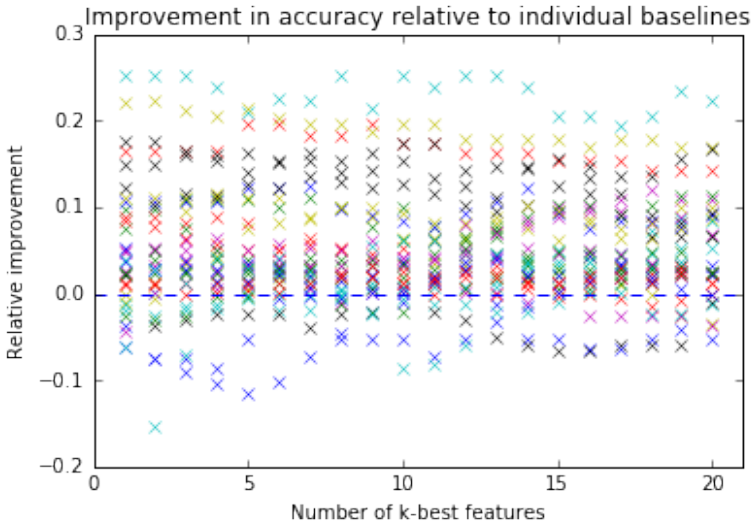

Figure 4.5: The distribution of classification results by training a Naive Bayes model on each individual subject using feature selection evaluated by F-value. Each row is the number of features selected, each x in each row is one subject. The relative accuracy is found by subtracting the individual baseline from the classification result.

questions where the answers are known in order to set an individual baseline. Given this, training subject-dependent models makes sense. One way to do this is to simply train individual classifiers on each speaker. This is the approach taken here.

The experiments from section 4.2 are repeated on individual speakers. Feature selection is done by taking the K-best features given F-value. χ^2 was left out as F-value was found to give better results.

Figure 4.5 shows the distribution of improvements over the individual baseline for each of the speakers over different numbers of features selected. Each individual speaker will have a different baseline as they lied different amounts. Some might be 70% or 80% truthful while others are closer to a 50/50 split. Therefore I have simply subtracted the individual majority class baseline from the classifier's cross-validation accuracy.

The interesting thing to note is that the majority of models classify better than the baseline, some even being as much as 20% (0.2 in the graph) above the baseline. In figure 4.6 we see the average accuracy over all subjects. The average score is far above the baseline. The best result is 66.0%, which is the best experimental result we have found and it only uses a small number of features. This shows that subject-dependency is highly effective and further experiments are likely to pay off.

Table 4.3 shows the best features for eight different subjects. As we can see these vary drastically from subject to subject. The specific words used are different, supposedly because the stories they tell are different and because of their individual speaking patterns. We see sentiment scores and part of speech tags being important features although unigrams and bigrams still seem most important and make up the majority of chosen features.

Subject	Features
S-1A	absolutely, did, didnt, didnt know, dont, excellent, going, just, lg, mhm, things, time, um, yes, compound, 'sentence length', cc, nn, nnp, vbg
S-5A	'average word length', did, mhm, section, uh did, um, 'kay, 'combined sentiment score', 'positive sentiment', 'sentence length', cc, jj, nns, prp, rbr, uh, vb, vbn, vbp, 'filled pauses count'
S-2B	big, camping, did, did excellent, excellent, nd, new york, really, texas, things, ve, yes, york, 'negative sentiment', 'neutral sentiment', fw, nnps, vbg, wdt, wrb
S-6B	city, columbia, just, just like, kansas, kansas city, kind, know like, lg, like know, missouri, people, sort, things, yeah, 'neueutral sentiment', 'positive sentiment', cc, vbn, wp
S-3C	big, columbia, did, going, like, lived, new jersey, poor, sentencedisfluency, uh worddisfluency, ve, ve sentencedisfluency, years, 'combined sentiment score', 'positive sentiment', dt, md, nn, vb, wrb
S-7C	cause, excellent, good, kids, lg lg, little kids, pretty, pretty good, sang, say, want, worddisfluency, yeah, 'combined sentiment score', 'neutral sentiment', 'positive sentiment', jj, jjs, rb
S-4D	alright, did, dont know, good, mean, mhm, mhm uh, pretty, remember, sort like, uh did, um just, went, wine, prp, rb, vbd, wp, wrb, 'number of words'
S-8D	actually, gosh, just, know just, laugh, lg, lg lg, lg um, new york, probably, sentencedisfluency know, sorry, start, steak, uh, uh worddisfluency, um lg, york, 'positive sentiment', nnp

Table 4.3: The 20 best features for eight subjects. There are four different profiles that could be given to a subject which will change what tasks they need to lie on. These are indicated by letters A-D. The words in quotation marks are special features such as 'positive sentiment'. The rest are n-grams or part of speech tags.

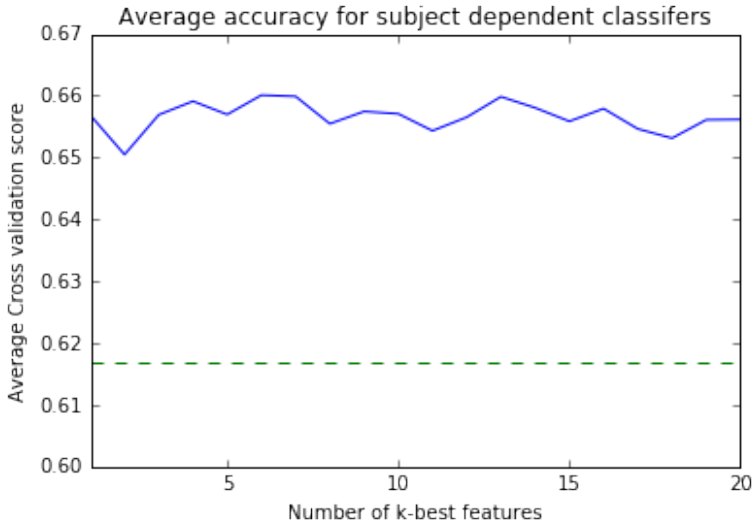

Figure 4.6: The average accuracy over all individual Naive Bayes classifiers for different numbers of features. The baseline is the average of individual majority class classifiers, which is the same as the baseline of the full dataset.

4.6 Summary

Five experiments were performed. The features were first tested using a simple Naive Bayes approach. Only two classifiers were better than the chance baseline. These only used the sentence length and number of words features and classified at 61.85%. Using the full feature set did not perform well. To make an improvement, feature selection was applied. Choosing four features and using a Naive Bayes classifier gave 62.5% accuracy. Further feature selection was applied and switching to Random Forest classifiers increased the result to 65.9% using 27 n-gram features.

To explore the impact of the prior on classification an experiment was run that artificially set and shifted the prior of the Naive Bayes models. For the full feature set it was found that a larger prior in favour of truth increased accuracy as the classifier was brought closer to the majority class classifier. For a Naive Bayes using only the four best features, using a prior close to the one trained by the model gave the best results. It looks like the majority of instances are being classified using only the prior as the results change drastically when going over the 0.5 prior point.

Finally, individual classifiers were trained for each subject using the Naive Bayes feature selection technique. Training and testing on a single subject improved results drastically. The average subject-dependent classification accuracy was 66.0% at the best point, far above the best Naive Bayes classifier on the full dataset that scored 62.5%. The most important features were very different from subject to subject, indicating that lying manifested itself in different ways with different people. For a summary of the best results compared with the best results in the literature see table 4.4.

Classifier	Features	Subject-Dependent	Result
Lexical DT	V		+0.8
Combo SVM	V+NV		+3.8
Combo-SD	V+NV	Yes	**+6.2**
NB+FS	V		+0.8
RF+FS	V		**+4.2**
SD-NB+FS	V	Yes	+4.3

Table 4.4: The best results on the CSC. The first three are benchmarks from literature. They are A lexical decision tree, an SVM using a combination of lexical, prosodic, and acoustic, and a subject-dependent combination of all feature types. Next we have Naive Bayes with feature selection, random forest with feature selectioin, and subject-dependent Naive Bayes. Features indicates what features used, either verbal (V) or non-verbal (NV). The scores are given as improvements over the baseline as the baseline recorded by Hirschberg et al. is slightly lower than what I found.

Chapter 5

Discussion

5.1 Technical Discussion

The first thing we can note is that this problem is difficult. We might have hoped to get some indication of positive results by using a simple model like Naive Bayes together with bag-of-words, however it almost seems that the more features we add the worse classification gets. In fact, if there is too much information, as seems to be the case with the full feature set and when adding too many features in the feature selection experiments, the best response seems to simply prefer the majority class, as indicated by the first shifting prior experiment. We saw that this allowed our classifier to do slightly better than chance when using a high prior. What seems to happen is that truth is preferred in the majority of cases except for a few where the algorithm is certain enough to say that it is a lie. This strategy seems to be mirrored in the much better classifier from the experiment in section 4.2 that uses feature selection. When shifting prior is applied to this experiment we see that the majority of instances are only being classified by the prior and lies are only chosen if the classifier is certainty that it is a lie. This is decided by the chosen features, or deceptive cues.

This is similar to the approach people have taken to deception detection, they look for a few specific cues that they believe indicate deception. However, the cues they choose are often completely incorrect and may even correlate with truthfulness [21]. In machine learning this phenomenon would be equivalent to overfitting. The learning algorithm has learned from noise in the data and thus associates a feature with deceptiveness that may not be correlated in general. It is quite likely that this is what is happening with the full feature set classifier. There are only a small number of separate instances in the dataset (4134). With the full feature set there are 3343 features. Fitting 3343 probability distributions using 4000 examples, all of them containing sparse bag-of-word models, is bound to be difficult. The estimated probabilities will be bad which will lead to a bad model. The reason feature selection is so successful is that the features that are overfitting can be removed and only the real deceptive cues are kept. In section 4.2 we do see that adding more features often reduces the accuracy of the classifier, which is no doubt due to overfitting.

What should be said, however, is that even with the simple Naive Bayes four feature classifier the same improvement over the baseline was gained as the lexical features of Hirschberg et al. Their lexical features give a cross-validation accuracy 0.8% over the baseline. We go on to increase this to an improvement of 4.2% using further feature selection and a Random Forest classifier. This result is better than previous work on lexical features by over 5 times and is better than the best published result for a subject-independent classifier which was 3.8% better than the baseline and used a combination of lexical, prosodic, and acoustic features. This bodes well for future work where acoustic and prosodic features can be added to further increase the performance of the classifier. Looking at the features used by the classifier we see that n-gram counts were the only features used. This supports what was found in work such as Mihalcea et al. [33] that found n-gram bags of words to be as good or better than LIWC.

5.1.1 Critique of Random Forest Feature Selection

The results of the Random Forest and feature selection experiment (section 4.4) should be taken with a grain of salt. The features are selected based on the overall improvement on average cross validation score, which is the value reported. What may be happening is that the greedy approach to feature selection may only work for a specific split of the data. It is very likely that many features are not indicative of deception in general, but are simply specific to the CSC data, due to the stories told by subjects or the questions asked. This means that the classifier is essentially overfitting to the entire dataset by picking features that are specific to some subset but does not impact the whole dataset. An example of such a feature is “violin”. It is highly unlikely that there is anything inherently deceptive about the word “violin”, what is more likely is that there is one story being told by a subject where she speaks of violins and in all cases they are lies. Thus the only time violins come up in the dataset is in lies. It should be trivial to see that a perfectly valid approach to increasing cross-validation accuracy is to start with a majority class classifier and simply find any features that appear only in lies and use these as features. Some of these may actually be indicative of deception while some, like violin, are most likely not.

That being said, we can still look at the features that have been chosen. We saw in the literature review that intensifiers like “really” and and assertion “exactly” could be correlated with deception. Thus it is not surprising to see these among the selected features. Features such as “poor” and “did really” may be domain specific features, in the interviews subjects were asked about their performance of the tasks and would answer in terms of “good” or “poor”. The way they answer these questions is very significant for this task, so someone who says “did really poorly” or “did really well” may be trying to intensify which may indicate deception.

5.1.2 Subject-dependent classification

The best published result was found by using subject-dependent features [21]. This is supported by my experiments. Even though the experiment run is simple, using only a

low number of features, the results are better than those found by the Random Forest and feature selection. They are not as good as those given by Hirschberg et al. but their classifier uses acoustic and prosodic data. One problem with the current approach is that separate classifiers are trained on each subject. Finding deceptive data for new subjects is unlikely, so this approach will not work. Hirschberg et al use subject-dependent features, which may be possible to extract for new speakers. Different ways of adapting to new speakers are discussed in Chapter 6 on future work.

It is interesting to note that different people are easier and harder to classify for deception. Some have improvements that are 20% over the baseline while others are 10% or more below the baseline. This indicates that the way people lie is very different and that some may not even show many signs of deception at all, while others are very obvious. We could look at it as having found some people's "tell" while others have a stone cold poker face. The features selected by the feature selection algorithm are also pointing towards people lying differently to each other as most of the features are very different from subject to subject. This shows that incorporating some kind of subject-dependency is very important for deception detection. Even the polygraph seems to agree with this as control questions are asked and an individual baseline is set before questioning.

5.2 Socio-technical considerations

Although some encouraging results have been shown it should be pointed out that this problem is far from solved. None of the work that has been reviewed indicates that we are close to a comprehensive machine learning approach to general deception detection. Some results are encouraging, with many results being significantly better than the the recorded baselines [40, 33, 30], but some points must be considered before deception detection software is released. Currently we are seeing results that are better than human judgement in many cases, although this is not difficult as many human judges perform below chance. This does not mean we should release our software claiming that it is better than humans at detecting deception. We need to understand what impact it may have if it was used.

To understand what our goal is we must look at the context in which it may be used. The context will directly decide what the consequences of detected deception will be. For example, in the case of deceptive opinion spam we may be interested in building a spam filter. If the spam filter has 90% accuracy we will probably be quite happy to use it. Even if the accuracy on the new data is slightly lower than on the artificial dataset we can probably expect to reduce the amount of deceptive reviews, a few false negatives will not matter too much. No one will be hurt by such an outcome. Now, if we look at other proposed applications such as border control or interrogation the consequences are very different. A false positive may mean an innocent being sent to jail and a false negative might mean a criminal going free. The question is then what is more important? [13] mentions that a higher false positive rate may be acceptable if there are more true positives, that is higher recall over higher precision may be acceptable. I strongly disagree. In the justice system "innocent until proven guilty" is a governing

value. I think this should be held by deception detection systems. People will trust in these systems, so if we, in our design process, say that having a few more false positives is acceptable we will condemn a number of innocents. That is not acceptable.

Chapter 6

Future Work

The approach so far has focused on simple linguistic features. The best approach used only a few very specific words taken from the bag-of-words representation. One drawback of the bag-of-words vector is its sparsity. Fitting accurate probability estimates or decision rules for so many sparse parameters makes good training hard. Additionally, the bag-of-words model takes neither word similarity or sentence structure into account. Similar words such as "really" and "very" may have similar usage, however, our current model cannot take that into account. One way to deal with such a problem could be to extract more complex features, for example by using the LIWC program. However, we will propose using Neural Networks to learn complex features and word embeddings to represent words in a lower dimensional continuous vector space.

So far we have not used any features taken from the speech signal. In previous work the most successful approaches have heavily used speech features and the most successful approaches used a combination of both features. Two types of experiment would be done, first, creating a simple classifier using prosodic and/or acoustic features that could be combined with the current classifier. The second would be to integrate information taken from the speech signal to the neural network approach. These are discussed in section 6.2.

Finally, we saw that different speakers benefit from using different features as their individual lying is different. Since we can imagine that this makes the approach generalise badly to new speakers. If this is the case it would be helpful if we could make adaptations to our model for new features, taking into account their unique style. In section 6.3 I discuss potential ways of solving this problem and experiment that may confirm the suspicion that models will generalise badly to new subjects.

6.1 Neural Networks

Neural network (NN) approaches for everything from machine vision [29] to speech processing [20] to biology[1] have been both popular and very successful. The field of

[1] http://blog.kaggle.com/2012/11/01/deep-learning-how-i-did-it-merck-1st-place-interview/

paralinguistics has also used NNs to detect emotion [56]. They have been shown to be effective for combining several channels of data in a single architecture [56], which we can use to combine transcription and speech signal. The first step to this end would be to use a NN approach for the transcribed speech.

Currently the best features are individual words and bigrams. As we have seen, these can be difficult to train and tend to cause overfitting, they also suffer from ignoring context and word similarity. In NLP the use of continuous vectors to represent words has become very popular [32, 53, 7, 8]. The idea is that these vectors capture semantic and syntactic information, similar words will be close together in the vector space, so "house" and "home" would be close together while "house" and "time" are further apart. These have been used as of various NN systems for NLP tasks from POS tagging [7] to Machine Translation [6]. One nice thing is that a phrase representation can be built using these vectors by combining a sentence using Recurrent NNs (such as Long Short-Term Memory (LSTM) networks) [6] or Recursive NN [51]. For example, machine translation has been done end to end by simply creating a phrase representation of one language using Recurrent NNs and decoding the sentence with a second Recurrent NN that uses the phrase representation as part of the input [6]. I propose an approach where a phrase representation is created and fed as input to a classification network with a sigmoid output layer.

One approach for building the phrase representation is by using Recursive Neural Networks. In an interesting paper Socher et al. [51] use semi-supervised recursive autoencoders to create phrase embeddings that learn sentiment information. The sentiment representation is learned all the way down to the word embedding level, normally word embeddings learn to model some level of syntactic and semantic similarity by being trained on the unsupervised task of language modelling [7]. Interesting properties are learned this way, for example the distance between 'man' and 'woman', in word embedding space, is roughly the same as the distance between 'king' and 'queen'. Socher et al.'s approach learns to also include sentiment information, while words such as 'good' and 'bad' are close together in the standard approach, using Socher et al.'s approach pushes these words apart to represent their difference in sentiment. This is done by adding an error function to the learning algorithm that represents the error of the supervised sentiment analysis task. It would be very interesting to see if this approach could be used to create word and phrase representations that take into account deception information. This would be difficult to verify, however, because specifically deceptive words are not known and probably do not exist. Perhaps such words can be found by looking at which words are pushed further apart in the learning process.

A different, but similar, approach is that of Le and Mikolov [28]. They too propose a framework for learning representations for phrases or longer segments such as paragraphs. Their approach starts with word embeddings trained for language modelling on unlabelled text. A paragraph embedding is then trained, which is simply a real valued vector that is meant to learn the context or topic of the paragraph. These are trained by using language modelling and are treated as just another word embedding. Once this paragraph vector is trained it can be used as input to any machine learning algorithm, including other neural networks. One important point to make is that for new data these paragraph vectors are learned at test time using gradient descent. The paragraph

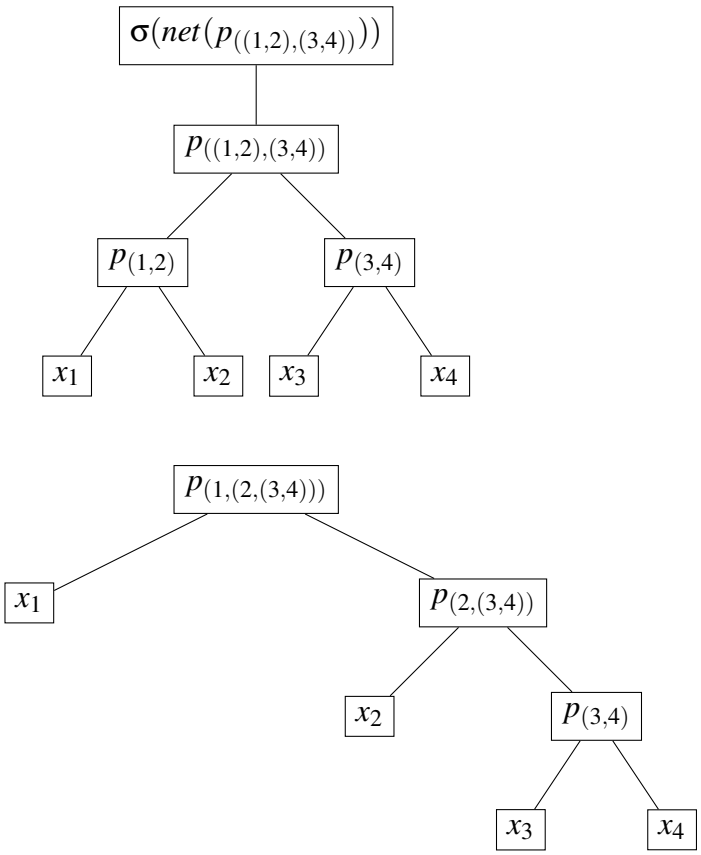

Figure 6.1: Here are two examples of recursive neural networks. Each leaf x_i is a word represented as a real valued vector. The network is applied on two words creating a new representation $p_{(i,j)}$. This can then be used as input together with a new word or phrase representation. The final representation is passed to an output network with a sigmoid output layer represented as $\sigma(\cdot)$ in the first network. The recursion can be applied in different ways. A heuristic search function can be used to pick the best tree.

representation could be used in the same way that Socher et al.'s phrase embeddings would, as input to a further Neural Network with a sigmoid output network.

One of the benefits of using a neural network would be that the output could be seen as an estimated probability which the output can be manipulated like a probability. This could allow us to repeat the shifting prior experiment on this new probability estimate. We will treat the estimate as being the posterior probability of the data belonging to a class. We can divide this by the class prior that the data would estimate, we can assume that this is the maximum likelihood estimate of the prior, that is the number of instances of a class over the total number of instances. If we divide the posterior by the prior we get a "scaled likelihood". It is not quite the likelihood as it is still divided by the probability of the data, but it will be proportional to the actual likelihood. We can then multiply in our own prior which we can set to what ever we like. This means we can influence the classifier in a way where we can go towards our goal of higher precision. Additionally, we could change the threshold at which we pick an instance to be a lie. Normally we would use 0.5 to be the threshold, however 50% certainty that someone is lying may not feel like enough. We could instead use a higher threshold

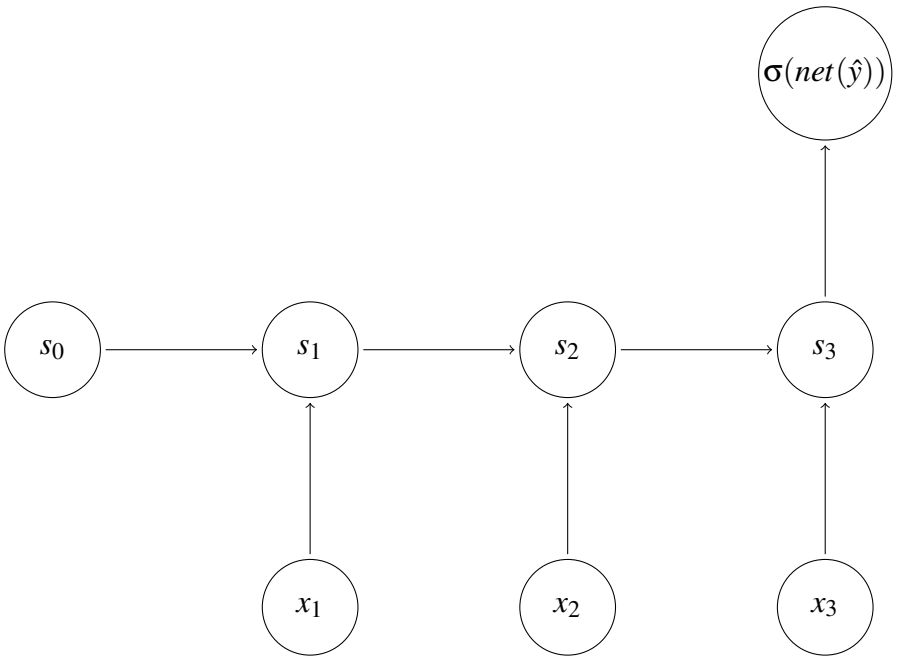

Figure 6.2: In a recurrent neural network the networks are applied to each input in turn. The network takes the previous state and the current input and creates a new hidden state that is passed forwards. The final hidden state is treated as a representation of the whole sentence and can be passed on to an output network. In binary classification the output layer will be a sigmoid function represented by $\sigma(\cdot)$.

forcing us to be 80% certain, for example. We can pick the ideal threshold by shifting this threshold and and finding the ideal value. These are benefits that a discriminative classifier like Random Forests do not have.

6.2 Speech features

The majority of success has come from using acoustic and prosodic features [21, 15, 2]. In this work we have left them out completely but we plan to incorporate them in future work. By simply replicating previous work we can expect to make an improvement over the current classifier as we have seen that the best results have come from combinations of classifiers for different feature types, for example through voting. The exact voting procedure would have to be decided upon, for high precision, a voting procedure where both classifiers must vote in favour of lying would probably be best, although using only one vote in favour could increase recall.

There would be several ways that the speech signal could be incorporated with the text neural networks. The first option would be to create a completely separate network. This network would output its own probability estimate for the two classes. We could then take a weighted average of the two networks as the actual probability, these weights could be tuned depending on how powerful each network is. One of the main network types that has been used for speech tasks is LSTM networks [20, 56]. These

networks allow variable length sequences of input to be used as input, like a speech sequence. The networks learn to use long term context by "remembering" through a hidden state representation. They work very similarly to Recurrent Neural Networks (see fig 6.2) but use a special hidden unit that has three gates that control what goes in and out of the hidden state, or "Memory Cell" [22].

A different approach could be to incorporate the speech signal in the network architecture. For example, for each word that goes in to the neural network the word embedding could be combined with a speech signal embedding. This embedding could, for example, be trained by applying a recurrent network on the the speech signal in the window between the start of the word and the end of the word. The output can then be combined with the word embedding to create a word embedding that "knows" how the word was said, as well as what the word was. A similar approach could be to extract acoustic and prosodic features for each word using software such as openSMILE and feed that as input to the network together with the words.

6.3 Speaker Adaptation

As mentioned in the discussion, individual speakers have different speech patterns and deceptive behaviour. Speaker-dependent models were found to improve classification greatly, however, these models cannot easily be used on new data as each model is trained on a single speaker and data tagged with deceptive content cannot be expected to be available for new subjects. Speech processing faces a similar problem where speakers have different pronunciations of the same words [43]. Approaches to deal with this problem are available for Gaussian Mixture Models which have been the most common systems in the past [17]. As Neural Networks have become more common for speech recognition research into adapting neural networks to new speakers has arisen [43, 31, 57]. It would be interesting to see if one of these approaches could be adapted to deception detection. I propose two possible speaker adaptation frameworks created for automatic speech recognition that may be able to be adapted to deception detection: retraining part of the network using test data or adaptation data as training examples and using i-vectors to automatically adapt to speakers.

For the first approach people generally have a set of adaptation data [58]. This can either be from a text someone has read out where the words are known, which means it can be treated as labelled data, or from unlabelled data. The labelled case could potentially be used for deception detection, control questions could be asked where the answer was known and the network adapted to that data which would be similar to the control questions in a polygraph test. However, this may lead to similar problems as those faced by a polygraph, where cheating on the control questions can allow you to cheat the whole test. In the unlabelled case the answers given by the network could be used as labels for retraining. This approach may also run into trouble as the network might simply learn to be more certain about its mistakes. Regardless of what input labels are used the network must be retrained. Retraining the whole network would not be possible as millions of parameters might need to be changed and it would be more prone to overfitting. Instead a linear layer can be added either after the input,

before the output layer, or after some hidden layer [58]. This layer would be the only part to be retrained.

The second approach may be easier to implement successfully as it does not depend on labelled data. The approach uses an identity vector, or i-vector, to identify each subject. The i-vector is built in an unsupervised manner by using the Expectation Maximization (EM) algorithm, and are concatenated to each input [43]. Since these are present at training time the learning algorithm can learn speaker dependencies from the start. New i-vectors can be estimated for new speakers which means testing is automatically adapted to new speakers. This idea is easily adaptable to neural networks as it simply makes the input slightly bigger [43]. Additionally, it has already been used on paralinguistic tasks such as cognitive load detection [50].

6.4 Using text given from ASR rather than transcriptions.

So far hand labelled transcriptions have been used for the text features. These are good because they are clean and allow for easy features extraction. They come with the added benefit of being aligned to the speech which means meaningful features can be extracted for individual words by using the time alignment. However, an ideal situation like this may be quite unlikely in practice. Perhaps it would be possible that a transcription was available in a court room or after a police interview and these could be force aligned to the speech and then analysed by the deception detection system, however, in many cases none of this would be available, most likely the only data available is the raw speech. Ideally, the algorithm should be able to deal with this. Since we are relying on text features we would have to extract them straight from the speech signal, meaning we would have to run the data through an ASR system.

If time allows I would like to run the speech through a speech recogniser. This data could be used for cross-validation, we could train on 90% of the transcribed data and test on the remaining 10% but using the ASR output. The other option would be to simply train and test on the ASR output. If neural networks are being used pretraining with denosing autoencoders [54] might help deal with the noise that will come from bad ASR output. Doing this for text input, which is discrete, might be difficult as in a one-hot encoding each word will be orthogonal to each other word. Therefore trying to model the fact that some word often get mapped to a different word may be difficult. Perhaps instead of a one-hot encoding a more fluid word vector could be used with probabilities that a given input is a certain word. Denoising could also be applied at the word embedding level by trying to map the incorrect word encoding to the correct one. These are simply speculations and rigorously testing these ideas may go beyond the scope of this project.

6.5 Interspeech deception data

In the Interspeech 2016 challenge one of the sub-tasks is deception detection [45]. The Deceptive Speech Corpus (DSC) is provided which was gathered using a "mock-crime" scenario. Students were recruited to go into an office and either steal an exam key or do some trivial task. They were then interviewed about the situation. The interview had two stages, a control stage where everyone was expected to speak the truth and the second stage where all thieves were expected to lie on all questions. This creates a scenario which can be treated as truth/lie classification.

One thing that we would wish for our models is that they should generalise well. We have looked at what this means for subject dependence. In addition to generalising well to new subjects we would hope for our approach to generalise to different domains and different data. I propose using the data in two different ways.

First, in deceptive opinion spam detection they found that a model trained on one type of review performs badly on other types. This is quite likely to be true for speech as well. To test this theory I would test the model created on the CSC corpus on the Interspeech data. Additionally, it would be interesting to see if the speaker adaptation approach could be used to also do "data adaptation". That is, to see if the algorithm could be made to adapt to a new deception scenario. All of this would simply take the form of training a model on the CSC corpus and testing on the DSC.

The second type of generalisation we may be interested in is simply to see if the model can be trained on a different domain and still do well. This is more about if the features, classifiers, or network architecture has been created in such a way that it only works for one type of data and not for others. The experiments would simply be training and testing the model on the DSC. The challenge has a defined training and validation split, so this can be used as train and test sets.

One problem with this data is that it is not transcribed. This means the only way to get lexical features would be to use output from an ASR system.

6.6 Summary and Priority Plan

The current work is incomplete, we are only using one channel of the available data ignoring the speech signal completely. Next year the first priority will be to make use of this vital source of information. The best results have always incorporated both text and speech signal [21, 15]. However, the work on lexical or linguistic features is not done. Neural Network have been successful in many applications and deception detection will no doubt benefit from their usage. The approaches proposed include heavy use of pretraining on unlabelled data. This can often take weeks when training is done on millions of words [8]. For this reason setting up these models to start training the unsupervised parts must also be given high priority. Once these threads have been explored separately they can be combined by incorporating prosodic and acoustic

information in the Neural Network models or by creating a new neural network model purpose built for the combination of the two channels.

Beyond this, further work will have to do with trying to generalise the approach in different ways. Creating speaker adapting model would bring the power of subject-dependent classifiers to new data, using ASR output instead of transcriptions would allow for a fully automatic model that could go from any speech input to deception detection, finally, the generalisation and adaptability of the algorithm can be tested on the DSC.

Chapter 7

Conclusion

Some simple ways of using lexical features other than those extracted from the LIWC program have been explored on the CSC corpus of deceptive speech. Initial models using thousands of features were found to perform badly in cross-validation experiments. Feature selection was found to make radical improvements and a final model using 27 n-gram counts as features ended up performing far above the previous best result for lexical features, the Random Forest classifier had 65.9% accuracy, 4.2% above the baseline. This is 5 times more than the previous best lexical only classifier that made a 0.8% improvement over the baseline. An experiment into subject-dependent classification supported the claim that subject-dependent classification can make large improvements over subject-independent classification by creating individual classifiers for each subject. The average classification accuracy for these, using only four features and a Naive Bayes classifier, was 66.0%. An extensive further work section proposes Neural Network approaches for text text classification of deception as well as ways to incorporate acoustic and prosodic information in such such models. Additionally, speaker adaptation approaches are discussed. The use of these may allow for the power of speaker-dependent models to be used on new data.

Bibliography

[1] Michael G Aamodt and Heather Custer. Who can best catch a liar? a meta-analysis of individual differences in detecting deception. *Forensic Examiner*, 15(1):6–11, 2006.

[2] Stefan Benus, Frank Enos, Julia Bell Hirschberg, and Elizabeth Shriberg. Pauses in deceptive speech. Columbia University Academic Commons, 2006. http://hdl.handle.net/10022/AC:P:20856.

[3] Steven Bird. NLTK: the natural language toolkit. In *ACL 2006, 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, Sydney, Australia, 17-21 July 2006*, 2006.

[4] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[5] Gokul Chittaranjan and Hayley Hung. Are you awerewolf? detecting deceptive roles and outcomes in a conversational role-playing game. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2010, 14-19 March 2010, Sheraton Dallas Hotel, Dallas, Texas, USA*, pages 5334–5337, 2010.

[6] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[7] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[8] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

[9] Bella M DePaulo, James J Lindsay, Brian E Malone, Laura Muhlenbruck, Kelly Charlton, and Harri Cooper. Cues to deception. *Psychological bulletin*, 129(1):74, 2003.

[10] Paul Ekman. *Telling lies: Clues to deceit in the marketplace, marriage, and politics*. New York: Norton, 1985.

[11] Paul Ekman. Why don't we catch liars? *Social research*, pages 801–817, 1996.

[12] Paul Ekman and Wallace V Friesen. Nonverbal leakage and clues to deception. *Psychiatry*, 32(1):88–106, 1969.

[13] Aaron Chaim Elkins. *Vocalic markers of deception and cognitive dissonance for automated emotion detection systems*. PhD thesis, The University of Arizona., 2011.

[14] Frank Enos, Stefan Benus, Robin L. Cautin, Martin Graciarena, Julia Hirschberg, and Elizabeth Shriberg. Personality factors in human deception detection: comparing human to machine performance. In *INTERSPEECH 2006 - ICSLP, Ninth International Conference on Spoken Language Processing, Pittsburgh, PA, USA, September 17-21, 2006*, 2006.

[15] Frank Enos, Elizabeth Shriberg, Martin Graciarena, Julia Hirschberg, and Andreas Stolcke. Detecting deception using critical segments. In *INTERSPEECH 2007, 8th Annual Conference of the International Speech Communication Association, Antwerp, Belgium, August 27-31, 2007*, pages 2281–2284, 2007.

[16] Florian Eyben, Martin Wöllmer, and Björn Schuller. Opensmile: the munich versatile and fast open-source audio feature extractor. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1459–1462. ACM, 2010.

[17] M. J. F. Gales. Maximum likelihood linear transformations for hmm-based speech recognition. *Computer Speech & Language*, 12(2):75–98, 1998.

[18] Gábor Gosztolya, Tamás Grósz, Róbert Busa-Fekete, and László Tóth. Detecting the intensity of cognitive and physical load using adaboost and deep rectifier neural networks. In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pages 452–456, 2014.

[19] Martin Graciarena, Elizabeth Shriberg, Andreas Stolcke, Frank Enos, Julia Hirschberg, and Sachin S. Kajarekar. Combining prosodic lexical and cepstral systems for deceptive speech detection. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing, ICASSP 2006, Toulouse, France, May 14-19, 2006*, pages 1033–1036, 2006.

[20] Alan Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. Hybrid speech recognition with deep bidirectional lstm. In *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pages 273–278. IEEE, 2013.

[21] Julia Hirschberg, Stefan Benus, Jason M. Brenier, Frank Enos, Sarah Friedman, Sarah Gilman, Cynthia Girand, Martin Graciarena, Andreas Kathol, Laura Michaelis, Bryan L. Pellom, Elizabeth Shriberg, and Andreas Stolcke. Distinguishing deceptive from non-deceptive speech. In *INTERSPEECH 2005 - Eurospeech, 9th European Conference on Speech Communication and Technology, Lisbon, Portugal, September 4-8, 2005*, pages 1833–1836, 2005.

[22] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[23] Mark Huckvale. Prediction of cognitive load from speech with the VOQAL voice quality toolbox for the interspeech 2014 computational paralinguistics challenge. In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pages 741–745, 2014.

[24] Clayton J. Hutto and Eric Gilbert. VADER: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the Eighth International Conference on Weblogs and Social Media, ICWSM 2014, Ann Arbor, Michigan, USA, June 1-4, 2014.*, 2014.

[25] William G Iacono. Forensic lie detection procedures without scientific basis. *Journal of Forensic Psychology Practice*, 1(1):75–86, 2000.

[26] How Jing, Ting-yao Hu, Hung-Shin Lee, Wei-Chen Chen, Chi-Chun Lee, Yu Tsao, and Hsin-Min Wang. Ensemble of machine learning algorithms for cognitive and physical speaker load detection. In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pages 447–451, 2014.

[27] E Kastor. The worst fears, the worst reality: For parents, murder case strikes at the heart of darkness. *The Washington Post*, page A15, 1994.

[28] Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 1188–1196, 2014.

[29] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[30] Jiwei Li, Myle Ott, Claire Cardie, and Eduard H Hovy. Towards a general rule for identifying deceptive opinion spam. In *ACL (1)*, pages 1566–1576. Citeseer, 2014.

[31] Hank Liao. Speaker adaptation of context dependent deep neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 7947–7951, 2013.

[32] Thang Luong, Richard Socher, and Christopher D Manning. Better word representations with recursive neural networks for morphology. In *CoNLL*, pages 104–113. Citeseer, 2013.

[33] Rada Mihalcea and Carlo Strapparava. The lie detector: Explorations in the automatic recognition of deceptive language. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 309–312. Association for Computational Linguistics, 2009.

[34] Kevin Mitnick. *Ghost in the Wires: My Adventures as the World's Most Wanted Hacker*. Little, Brown, 2011.

[35] Kevin D Mitnick and William L Simon. *The art of deception: Controlling the human element of security*. John Wiley & Sons, 2011.

[36] Gelareh Mohammadi, Alessandro Vinciarelli, and Marcello Mortillaro. The voice of personality: Mapping nonverbal vocal behavior into trait attributions. In *Proceedings of the 2nd international workshop on Social signal processing*, pages 17–20. ACM, 2010.

[37] Claude Montacié and Marie-José Caraty. High-level speech event analysis for cognitive load classification. In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pages 731–735, 2014.

[38] Matthew L Newman, James W Pennebaker, Diane S Berry, and Jane M Richards. Lying words: Predicting deception from linguistic styles. *Personality and social psychology bulletin*, 29(5):665–675, 2003.

[39] Myle Ott, Claire Cardie, and Jeff Hancock. Estimating the prevalence of deception in online review communities. In *Proceedings of the 21st international conference on World Wide Web*, pages 201–210. ACM, 2012.

[40] Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T Hancock. 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*, pages 309–319. Association for Computational Linguistics, 2011.

[41] James W Pennebaker, Martha E Francis, and Roger J Booth. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, 71:2001, 2001.

[42] Verónica Pérez-Rosas, Mohamed Abouelenien, Rada Mihalcea, and Mihai Burzo. Deception detection using real-life trial data. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction, Seattle, WA, USA, November 09 - 13, 2015*, pages 59–66, 2015.

[43] George Saon, Hagen Soltau, David Nahamoo, and Michael Picheny. Speaker adaptation of neural network acoustic models using i-vectors. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding, Olomouc, Czech Republic, December 8-12, 2013*, pages 55–59, 2013.

[44] Björn Schuller, Stefan Steidl, Anton Batliner, Felix Burkhardt, Laurence Devillers, Christian MüLler, and Shrikanth Narayanan. Paralinguistics in speech and languagestate-of-the-art and the challenge. *Computer Speech & Language*, 27(1):4–39, 2013.

[45] Björn Schuller, Stefan Steidl, Anton Batliner, Julia Hirschberg, Judee K. Burgoon, Alice Baird, Aaron Elkins, Yue Zhang, Eduardo Coutinho, and Keelan Evanini. The interspeech 2016 computational paralinguistics challenge: Deception, sincerity & native language. ISCA, San Francisco, USA, 2016.

[46] Björn W. Schuller, Stefan Steidl, and Anton Batliner. The INTERSPEECH 2009 emotion challenge. In *INTERSPEECH 2009, 10th Annual Conference of the International Speech Communication Association, Brighton, United Kingdom, September 6-10, 2009*, pages 312–315, 2009.

[47] Björn W. Schuller, Stefan Steidl, Anton Batliner, Felix Burkhardt, Laurence Devillers, Christian A. Müller, and Shrikanth S. Narayanan. The INTERSPEECH 2010 paralinguistic challenge. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pages 2794–2797, 2010.

[48] Björn W. Schuller, Stefan Steidl, Anton Batliner, Julien Epps, Florian Eyben, Fabien Ringeval, Erik Marchi, and Yue Zhang. The INTERSPEECH 2014 computational paralinguistics challenge: cognitive & physical load. In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pages 427–431, 2014.

[49] Björn W. Schuller, Stefan Steidl, Anton Batliner, Alessandro Vinciarelli, Klaus R. Scherer, Fabien Ringeval, Mohamed Chetouani, Felix Weninger, Florian Eyben, Erik Marchi, Marcello Mortillaro, Hugues Salamin, Anna Polychroniou, Fabio Valente, and Samuel Kim. The INTERSPEECH 2013 computational paralinguistics challenge: social signals, conflict, emotion, autism. In *INTERSPEECH 2013, 14th Annual Conference of the International Speech Communication Association, Lyon, France, August 25-29, 2013*, pages 148–152, 2013.

[50] Maarten Van Segbroeck, Ruchir Travadi, Colin Vaz, Jangwon Kim, Matthew P. Black, Alexandros Potamianos, and Shrikanth S. Narayanan. Classification of cognitive load from speech using an i-vector framework. In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pages 751–755, 2014.

[51] Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 151–161, 2011.

[52] Committee to Review the Scientific Evidence on the Polygraph (National Research Council (US)), National Research Council (US). Board on Behavioral, Sensory Sciences, National Research Council (US). Committee on National Statistics, National Research Council (US). Division of Behavioral, and Social Sciences. *The polygraph and lie detection*. Haworth Press, 2003.

[53] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394. Association for Computational Linguistics, 2010.

[54] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research*, 11:3371–3408, 2010.

[55] Cynthia Whissell, Michael Fournier, Rene Pelland, Deborah Weir, and Katherine Makarec. A dictionary of affect in language: Iv. reliability, validity, and applications. *Perceptual and Motor Skills*, 62(3):875–888, 1986.

[56] Martin Wöllmer, Moritz Kaiser, Florian Eyben, Björn W. Schuller, and Gerhard Rigoll. Lstm-modeling of continuous emotions in an audiovisual affect recognition framework. *Image Vision Comput.*, 31(2):153–163, 2013.

[57] Kaisheng Yao, Dong Yu, Frank Seide, Hang Su, Li Deng, and Yifan Gong. Adaptation of context-dependent deep neural networks for automatic speech recognition. In *2012 IEEE Spoken Language Technology Workshop (SLT), Miami, FL, USA, December 2-5, 2012*, pages 366–369, 2012.

[58] Dong Yu, Kaisheng Yao, Hang Su, Gang Li, and Frank Seide. Kl-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 7893–7897, 2013.

[59] Miron Zuckerman, Bella M DePaulo, and Robert Rosenthal. Verbal and nonverbal communication of deception. *Advances in experimental social psychology*, 14(1):59, 1981.