Preparation for Thesis Project in spring 2019

by Rasmus Lehmann (rale@itu.dk)
supervised by Leon Derczynski

November 27, 2019

Contents

Abstract

Even for high-profile politicians, it is growing increasingly easy to get away with basing your opinions on postulates and half truths, simply changing your opinion the next day, of which the success of POTUS Donalt Trump would be an obvious example. For a democracy to function, it is crucial for individuals to be able to identify the opinions of their politicians, allowing informed voting. This makes it important to build systems which can help citizens navigate the political field. Here I present preliminary work, including a literature review of the field of classification of natural language corpora and an actionable project plan, leading up to a Thesis Project to be executed in spring 2019, which will present an approach to target-specific classification of politicians' opinions.

1 Introduction

The role of this paper is to present the reader with the scope and final goal of my thesis project to be executed in spring 2019, the project plan for this thesis project as well as the findings of the literature review performed for the course Thesis Preparation at the IT University of Copenhagen fall 2018, which has helped set the parameters of the thesis project. The paper leads with a presentation of the scale and scope of the thesis project, followed by the literature review and finally the thesis project plan.

Thesis project scope and goal As an individual, it can be difficult to keep track of the ever changing opinion of politicians, especially when an opinion stated by a politician one day, can be refuted the next, by the same person. Using statistics, it would be a simple task to calculate the stance of a politician on a given issue, simply based on how often they speak out for or against that issue. Such an approach faces a few challenges.

1. A dataset containing data over politicians' statements related to a number of relevant issues is required
2. The dataset would need to be manually maintained, if the analysis is to be continual, requiring a large amount of manual labor

The fact that such a dataset over politicians' statements does not currently exist for Danish, and therefore would need to be created and manually labeled for stance, combined with the labor-intensive nature of maintaining said dataset, makes it natural to look for a way to optimize these tasks. Using a machine learning approach, it would be possible to automatically classify politicians' stances based on a sub-dataset of manually annotated corpora, decreasing the manual labor requirements of both generating and maintaining such a dataset.

The goal of the thesis project to be executed in spring 2019 will thus be to

1. Generate and annotate a dataset over politicians' statements regarding a number of current issues
2. Build a machine learning model which allows the classification of politicians' stances towards said issues based on the generated dataset

Aside from the information this dataset and machine learning model would generate regarding politicians' opinions, both the dataset and the model would be

valuable contributions to the field of Natural Language Processing (NLP). The dataset, the first of its kind for Danish, would make it possible for researchers to train and test their Danish stance detection models, and the model proposed in the thesis project would make for a possible baseline on which to base evaluation of further stance classification efforts.

Literature review The goal of the literature review has been to grant an insight into the current methods in use when generating datasets for and handling tasks within classification in NLP. A broad view of the field was taken, including both tried and true approaches to NLP tasks such as the use of Support Vector Machines and Bayesian classifiers, and more novel approaches such as bidirectional Long Short Term Memory models and hierarchical model structures. This has resulted in a literature review which covers many different approaches to stance detection, rather than expanding deeply on a few of them.

Seeing as the goal of the literature review is to create a foundation of knowledge on which to make decisions regarding the approach chosen in the final thesis project, decisions regarding which directions to expand the review in have been made incrementally, as my knowledge of the field has grown.

2 Current Approaches in NLP Classification

In this section will be presented a number of implementations of solutions to NLP classification tasks, using a number of different classification models, including both neural networks and kernel-based classifiers.

2.1 Supervised learning approaches

Below are listed implementations making use of supervised learning. Supervised learning approaches cover approaches for which a test and training dataset has been manually labeled, thus ensuring the correctness of labels in the training and test data.

Bayes Classifiers [Qazvinian et al., 2011] apply Bayes classifiers in their work on rumor identification and belief classification, using a set of features focused on tweet-specific data. Four categories of features are identified

- **Content-based Features** includes Part of Speech (POS) and lexical patterns extracted from tweet text
- **Network-based Features** covers whether a user historically has tweeted refutal or support of rumors, and whether a user historically has re-tweeted other users who either support of refute rumors
- **Twitter-specific Features** includes a user's use of hashtags and URLs, often found in relation to either refutal or support of rumors

For each feature a Bayes classifier is built using two probabilistic models, one created from a set of supporting and one from a set of refuting tweets. These models are used in calculating the log likelihood ratio of a tweet being in the positive rather than the negative model, with respect to the given feature, calculated as follows, $P(\theta^{-}|t)$ representing the probability of a given tweet being in the negative model and $P(\theta^{+}|t)$ representing the probability of a given tweet being in the positive model

$$LL_i = log\frac{P(\theta_i^-|t)}{P(\theta_i^+|t)}$$

For optimization an L_1-regularized log-linear model is maximized using the features described above.

[Qazvinian et al., 2011] find misclassification of tweets as rumors to be a major issue in creating a clean dataset for rumor stance classification. They therefore seek to apply their model for classifying tweet relevance in relation to a given rumor, training a separate model for each of their feature groups to allow comparison of the feature types' effect on performance. They find that context-based features are the most influential on precision by a small margin, and on recall by a large margin, and combining all features yields a mean average precision of 0.965. For belief classification [Qazvinian et al., 2011] find that using only context-based features yields the highest accuracy and F-score, but using all features yields the highest precision.

Support Vector Machines (SVM) Implementing a linear-kernel SVM, [Mohammad et al., 2017] use sentiment analysis to increase efficiency of the stance detection models on the dataset used for the SemEval-2016 Task 6. Five models are trained on five separate datasets of tweets, based on tweet target. Sentiments of tweets are determined using five pre-defined lexicons and used as a feature in the SVM models. Besides sentiment, [Mohammad et al., 2017] make use of the following features.

- **n-grams** character and word n-grams of size 3, 4 and 5
- **Target** absence/presence of sentiment target in tweet
- **POS** occurrence count for each POS tag in tweet

- **Encodings** presence/absence of positive and negative emoticons, hashtags, punctuation, elongated words and upper-case characters

[Mohammad et al., 2017] manage to outperform the best implementations submitted for SemEval-2016 Task 6 using their SVM implementation and only n-gram features. Comparing possible feature combinations, [Mohammad et al., 2017] find that including all features decreases performance of their model, concluding that the optimal combination for this task is n-grams and sentiment. Furthermore, it is found that results can be improved further by using word2vec word embeddings instead of embeddings using random initialization.

[Enayet and El-Beltagy, 2017] similarly build a linear-kernel SVM for their entry to the SemEval-2018 Task 8 competition, applying a Naïve Bayes classifier for semantic feature extraction. Preprocessing consists of removal of stop words, punctuation and twitter keywords. Experiments are made with further preprocessing in the form of removal of URLs, stemming, lowercasing and generation of bi-grams, all of which reduces the performance of the model. [Enayet and El-Beltagy, 2017] extract a total of twenty features for each corpus. For subtask A, in which the model is to classify the stance of a given tweet towards a source tweet which is a rumor, the following features are included

- Question, denial, support and url in tweet
- Tweet is a reply
- Tweet sentiment
- Number of tweeter followers
- Source tweet user is verified
- User 'replied to' is verified
- Cosine similarity to root tweet

For this task, [Enayet and El-Beltagy, 2017] ranked 5 out of 8, and are furthermore outperformed by the baseline method of automatically classifying by majority class.

For subtask B, the model is set to classifying the veracity of a given rumor tweet, only based on tweet text, as either true, false or unknown, and give a confidence value between 0 and 1 for this classification. Here [Enayet and El-Beltagy, 2017] extract the presence of a hashtag and/or URL as binary features and the percentage of replying tweets that are classified as querying, denying and supporting. For this subtask [Enayet and El-Beltagy, 2017] outperform all competitors, although still falling below the most-common-class baseline.

Convolutional Kernels Applying a kernel-based tree approach using a voted perceptron to a parsing task, [Collins and Duffy, 2001] seek to show the capacity of kernel-based approaches for handling the high-dimensionality feature spaces inherent to NLP, and create a model that is generally applicable to other tasks within NLP. [Collins and Duffy, 2001] claim tree models to be computationally heavy due to the task growing exponentially with the number of sub-trees, nevertheless claiming to be able to complete such tasks in polynomial time using a convolutional kernel-based approach. Using importance-scaling of trees based on size at between 0.5 and 0.9, a probabilistic context-free grammar and a maximum depth of three for subtrees, [Collins and Duffy, 2001] achieve an average precision recall of 81 %.

Random Forests In a comparative study, [Zeng et al., 2016] analyze the performance of Random Forests for use in stance detection in rumors against a model using logistic regression and a model using Naive Bayes classification. All models are trained using the following features

- **Punctuation** including only exclamation and question marks
- **Twitter-element features** covering use of hashtags, URLs and reference to other users using twitter-handles (@)
- **LIWC** usage of negation and swear words, negative and positive emotion words, emoticons etc.
- **Sentiment features** found using the MetaMind sentiment classifier labeling tweets positive, negative or neutral
- **N-gram features** reduced applying stemming, lowercasing and a minimum frequency threshold
- **POS features**

The dataset is divided into five subsets each related to a specific rumor. A model is trained for each classifier for each subset, as well as a model for each classifier trained on the full dataset. The Random Forest models outperform the logistic regression and Naive Bayes models on accuracy and F-score for every subset as well as the pooled set. On recall the logistic regression models outperform the Random Forest models for a few of the subsets, and on precision the Naive Bayes outperforms the Random Forest on a few subsets. The ten most important features for the classification task are all found to be either n-grams or POS tags, the unigram *not* being the single most important feature, being six times as important as the bigram *not isis*, which is the second-most important feature.

Convolutional Neural Networks (CNN) [Kim, 2014] explores the applicability of CNNs for classification tasks within NLP. Prior to this effort, CNNs had been found highly effective for tasks within computer vision and speech recognition, but had yet to show promise for NLP classification. [Kim, 2014] constructs a simple CNN, representing sentences as a concatenation of word vectors, applying a filter window of 3, 4 and 5 respectively to generate filter maps, using max-over-time-pooling to extract the most influential feature. These are passed through a final fully-connected layer with dropout of 0.5 and a softmax activation function. [Kim, 2014] creates four slight variations on this model:

- **CNN-rand** using randomized initial word embeddings and training of word embeddings
- **CNN-static** using word2vec word embeddings but no training of word embeddings
- **CNN-non-static** using word2vec and allowing training of word embeddings for each task
- **CNN-multichannel** using two sets of word2vec embeddings, training one while keeping the other static

[Kim, 2014]'s implementations are tested across seven different classification tasks against a total of fourteen models, and manages to outperform all models on four tasks, despite competing against more complex models, in many cases with much more sophisticated preprocessing requirements, thus proving the effectiveness of CNNs for NLP classification tasks. Through a comparison of the four CNN variations, [Kim, 2014] finds that the word2vec embeddings significantly increases performance, that training word embeddings for the specific task generally increases performance and that the multichannel approach can not be shown to improve results.

Recursive Neural Networks (RNN) Applying a RNN to perform ideology and bias detection on a dataset of congressional debates, [Iyyer et al., 2014] seek to create an approach which takes sentence syntax and semantic composition into account, thus going further than the standard bag of words (BOW) and wordlist approaches of the time. Sentence vectors are generated recursively from the word-level going to phrase level and finally to sentences, using the function below, where f is a nonlinear activation function, X_a and X_b are word vectors, X_c is the phrase vector, b is a bias and W_{left} and W_{right} are composition matrices of word embeddings of size d*d, d being the size of word vectors.

$$X_c = f(W_{left} * X_a + W_{right} * X_b + b)$$

Sentence vectors are passed through a softmax layer, the results of which are optimized to minimize cross-entropy loss over all sentences in the training data. Hyperparameters, that is composition matrices, biases and the category-matrix used in the softmax function, are optimized using a breadth-first graph search. Three RNN models are built, one using random initialization of word embeddings and sentence-level labels, one using word2vec pre-trained word embeddings and sentence-level labels and one using word2vec embeddings, sentence-level labels and phrase-level labels. These models are tested against a number of linear regression models using BOW implementations, all of which are outperformed by even the simplest RNN implementation. It is found that the inclusion of phrase-level labels in the RNN improves performance significantly, especially for more complex sentence structures.

More recently, [Ma et al., 2018b] perform a comparative analysis of top-down and bottom-up tree-structured RNNs for rumor identification and rumor stance classification, seeking to create a model which takes both content semantics and structural information of their corpora into consideration - an approach which, according to [Ma et al., 2018b], has been largely overlooked within the field of rumor analysis using machine learning. A node in the RNN represents a single tweet, with the source tweet being the root node and conversation threads branching out from that.

[Ma et al., 2018b]'s models apply Gated Recurrent Units (GRUs), applying reset and update gates to retain information across deep conversation threads. The bottom-up and top-down models are relatively similarly structured, with one difference that for the bottom-up implementation, the hidden states of child-nodes are summed to allow element-wise multiplication, which is not applicable in the top-down model, where the hidden state of the mother node is used instead. For both models, a hyperbolic tangent activation function is applied, with a final softmax layer used for classification. For the bottom-up model softmax is applied to the source node, where for the top-down model a pooling function is applied to all leaf nodes thus generating an output vector on which softmax is applied.

Word embeddings are randomly initialized, and the model is trained to minimize the squared error using L_2 regularization. The models are tested on the quaternary classification task of classifying a given rumor in a source node based on it conversational tree as either non-rumor, false rumor, true rumor or unverified rumor. As baselines for comparison [Ma et al., 2018b] implement eight models, including three tree-based models, four SVM-based models and a single GRU-based RNN. Both the top-down and bottom-up model manage to outperform the baselines by a very comfortable margin, the top-down model outperforming the bottom-up model. It is worth noting, that the baseline-approaches

all rely on feature engineering, where the models proposed by [Ma et al., 2018b] do not, and the only features used for classification in the experiments are word embeddings.

A similar task of rumor identification and rumor stance classification is tackled by [Ma et al., 2018a] through the implementation of an RNN based on Multi-task learning. The two tasks are both viewed as supervised sequence classification problems, for rumor identification over the class labels non-rumor, true, false and unverified and for rumor stance classification over the class labels supporting, denying, querying and commenting. According to [Ma et al., 2018a], most prior approaches to rumor stance classification have treated the task as a single-instance classification based only on a single news headline or post. They seek to achieve stronger results for this task by taking full news articles and full post threads into consideration during classification. Two approaches to building RNN multi-task learning models are proposed.

The *Uniform Shared-Layer Architecture* uses a single RNN model and shares a GRU hidden layer between the two tasks, updating the hidden layer based on task-specific embeddings and input vectors. [Ma et al., 2018a] suspect that this approach might generalize the training of the model too much, making it unable to express that some patterns are more important to one task than the other. In the *Enhanced Shared-Layer Architecture*, a hidden GRU layer is similarly shared to extract common patterns, but a task-specific GRU is implemented for each task, and the two task-specific layers are connected to the shared layer with a weight matrix. Both architectures use a hyperbolic tangent activation function, and applies softmax for classification. For the uniform architecture, [Ma et al., 2018a] use the output of the shared layer as input for the softmax function, whereas for the enhanced architecture the output of the task-specific hidden layers are used as input for softmax. For both architectures, the only features applied are word embeddings.

[Ma et al., 2018a] train the models using stochastic gradient descent, randomly selecting the input task, and updating the model according to the task-specific objective. For benchmarking twelve models are implemented; four tree-based models, three SVM-based models, a naïve bayes classifier, a Hawkes Process classifier, a CNN, a bi-directional RNN and finally [Ma et al., 2018a]'s own RNN model with the stance classification module removed. For three separate datasets and both tasks, either the Uniform Architecture or Enhanced Architecture manage to outperform all competing models in regards to Macro-F1 and F1. The models are, however, outperformed in regards to Micro-F1 for every dataset and task. According to [Ma et al., 2018a], this is due to extremely strong performance on majority classes for e.g. the Random Forest classifier in their rumor identification task and the SVM-BOW implementation in stance classification.

Long Short Term Memory Models (LSTM) An approach to building a bidirectional LSTM is proposed by [Li et al., 2017], who use a CNN as the base of their LSTM, to increase the network's understanding of sequential correlation between words and sub-sentences. The model achieves bidirectionality by first reading a corpus from right to left and then left to right. [Li et al., 2017]'s CNN implementation takes word embeddings trained on tweets using Word2vec, as input, uses filter sizes of 2, 3 and 4 respectively, has a single convolutional layer and uses max-pooling to ensure a fixed-length vector. The output of the pooling layer, named the Target-related Representation by [Li et al., 2017], is concatenated with the original word embeddings and used as input for the LSTM

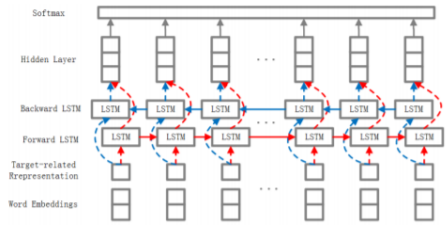

Figure 1: Convolutional Bidirectional LSTM model

Loss is calculated as cross-entropy error, and classification is performed using softmax.

SVM and Linear Regression models are set up as experimental baselines, both initiated using bag-of-words and Word2vec word representations. For comparison with other Neural Network approaches, [Li et al., 2017] implement two RNN models, three CNN models, one LSTM, one Bidirectional LSTM and one CNN-LSTM. The model is tested on five different datasets annotated for ideology classification, and manage to outperform all other implementations on 4 out of 5 datasets.

In their entry to the SemEval-2017 Task 8 competition, [Kochina et al., 2017] implement a sequential LSTM, making use of the structural information in tweet conversations to create a model which outperforms all competing systems. For preprocessing, all non-alphabetic characters are removed, text is lowercased and tweets are tokenized. The following features are extracted for each tweet

- Word vectors using Word2vec
- Tweet lexicon over negation- and swear words

- Presence of periods, exclamation marks, question marks and ratio of capitalized letters
- Presence of URLs and images
- Cosine similarity to source tweet, preceding tweet and whole tweet thread
- Content length
- Whether tweet is a source tweet

The final model applies a single LSTM unit with a single layer for each tweet, two ReLU layers, a 50 % dropout layer and finally a softmax layer to classify the tweet. The output vector of the LSTM is used as input state for the LSTM unit of the following tweet. Input vectors are generated by averaging word vectors and concatenating them with the additional features listed above. Experiments are performed using a bidirectional LSTM implementation as well as an LSTM implementation with lower granularity which takes word-level input at each time-step, both of which were found to decrease accuracy.

[Kochina et al., 2017] find that their model is not able to correctly classify denying tweets, and has issues classifying querying and supporting tweets, classifying most of these as commenting, the majority class. Nevertheless, the model outperforms all competitors for the SemEval-2018 Task 8 subtask A.

Active Learning [Skeppstedt et al., 2017] seek to implement a classification model for stance and sentiment detection, which is able to capture more complex representations of stance and sentiment than the current binary, ternary and quaternary classification approaches. This is done using seven parameters for which a given corpus is evaluated

- **Certainty** e.g. use of words like *definitely*
- **Uncertainty** e.g. use of words like *maybe*
- **Hypeotheticality** e.g. use of phrases like *if X had been*
- **Prediction** e.g. use of phrases like *X will be*
- **Recommendation** e.g. use of phrases like *X should never be*
- **Concession/Contrast** e.g. use of phrases like *X is good, but parts of X are bad*
- **Source** e.g. use of phrases like *According to X*

Unlike most supervised learning approaches where training data is extracted at random as a subset of the set on which a model is tested, [Skeppstedt et al., 2017] apply active sampling, extracting the training data which is deemed the most suitable for training their model. [Skeppstedt et al., 2017] implement a basic SVM model, and select the datapoints which are closest to the separating hyperplane of the classifier. Separate SVM models are trained for each of the seven parameters listed above, using unigrams and bigrams as features, initially using a seed dataset of just three datapoints for each model. Based on the classification hyperplane generated by these models, five additional datapoints are selected and annotated for each model, and the models are retrained. The approach is repeated until 2.096 sentences have been actively selected and annotated. Further manual annotation using a similar approach is done on chunk-level, to create features of higher granularity than those on uni- and bigram level.

Table 1: Results for three active learning methods

	F-score 1.525 training instances			**F-score** 2.096 training instances		
	Baseline	Granular	Cluster	Baseline	Granular	Cluster
Uncertainty	0.53	0.59	**0.63**	0.56	**0.61**	0.60
Certainty	0.29	0.50	**0.52**	0.36	**0.48**	**0.48**
Conc./Contrast	0.52	**0.66**	0.65	0.54	**0.67**	0.65
Hypotheticality	**0.65**	0.60	0.59	**0.66**	0.60	0.61
Recommend	0.48	**0.60**	0.57	0.51	0.55	**0.57**
Prediction	**0.50**	0.44	0.48	**0.54**	0.44	0.46
Source	0.35	0.31	**0.42**	0.35	0.37	**0.39**

[Skeppstedt et al., 2017] implement an alternative feature set to the two granularity-based features above by applying a clustering algorithm to Word2vec word embeddings for the words present in the training dataset. Clusters are found using the euclidian distance of aforementioned word embeddings, and a cluster representation vector is generated for each cluster, finally concatenating the cluster representation with the word embedding to generate a single feature vector. These input vectors are automatically generated, thus making the cluster-based approach weakly supervised.

Finally [Skeppstedt et al., 2017] compare a number of variations on their models, including the two variations on feature input listed above and models trained on the full training dataset of 2.095 sentences against models trained on a dataset of 1.525 sentences.

It was found, as shown in Table 1, that the strongest feature input is largely dependent on a combination of the size of the training dataset and the parameter for which the given sentence is being classified.

2.2 Weakly and distantly supervised learning approaches

In this section will be presented a number of implementations of solutions to NLP classification tasks, using weakly and distantly supervised learning. Weakly supervised learning approaches cover approaches for which a test and training dataset has been automatically labeled, for instance using a machine learning algorithm, thus not completely ensuring the correctness of labels in

test and training data. Weak supervision is often used alongside normal supervision. Distant supervision is the use of an already existing dataset to label the test and training data automatically.

Support Vector Machines (SVM) Extending their supervised SVM implementation with a weakly supervised variation, [Mohammad et al., 2017] generate additional training data by automatic labeling tweets based on manually labeled stance-indicative hashtags. It is found that this data is too noisy, and does not improve performance of the original SVM model. Further effort of enhancing the dataset is made by calculating associations between terms and stances using point-wise mutual information, calculated as:

$$PMI(word, label) = log\frac{freq(word, label) * corpusSize}{freq(word) * freq(label)}$$

These association features are found to increase performance of SVMs across all targets.

Process Specification Language (PSL) With a focus on classification using modeling tweets' social context, [Johnson and Goldwasser, 2016] seek to go beyond text-based classification. A dataset is generated consisting of 99.161 tweets from 32 American politicians equally split between the republican and democrat party. The following labels are applied on tweet-level

- **Tweet issue** using a keyword-issue mapping, a binary predicate is defined for each issue related to each tweet based on the majority matching keyword
- **Tweet sentiment** applying the ternary sentiment classification of [Wiebe et al., 2004]

The following labels are applied on politician-level

- **Content Agreement and Disagreement Patterns** is the frequency of similar word occurrences between two politicians, for a given issue
- **Temporal Activity Patterns** is a binary predicate showing whether two politicians tweeted about the same issue on the same day
- **Political Framing** using a keyword-framing mapping, the most used framing for a given issue is found on politician-level
- **Temporal Framing Pattern** similarly to activity patterns, a binary predicate is defined showing whether two politicians used the same frame for tweeting about a given issue on the same day

[Johnson and Goldwasser, 2016] apply these relation by specifying them as weighted logical formulas, defining three separate PSL models. The first classifies a given tweet based on the party affiliation of the tweeter, using the label most common for that issue, across tweets from all politicians in the same party. The second model includes whether the politician has tweeted positively or negatively about the issue earlier, and the third model includes patterns of content agreement, temporal activity, political framing and temporal framing. [Johnson and Goldwasser, 2016] implement a basic SVM as testing baseline, which is outperformed by all three PSL models. The test results show that the largest model, including all features mentioned above, gives the best performance, showing over 99 % accuracy for labeling on nine out of 32 subsets of the dataset.

Gaussian Naive Bayes [Lai et al., 2016] achieve strong results using an array of context-based features, excluding n-grams and word-embeddings entirely, resulting in the following list of feature types.

- **Sentiment-based Features** making use of a number of sentiment-lexicons and -dictionaries over e.g. negative/positive emotional words and polarity of words
- **Structural Features** including hashtags, punctuations and mentions of other users on the twitter platform
- **Context-based Features** includes mention of
  Stance target by name
  Stance target by pronoun
  Stance target's political party
  Opposition towards target's party colleague
  Stance target's opposition party
- **Label-based features** showing the overall sentiment of the tweet (positive, negative, neutral) and whether the stance target is present in the tweet

A number of models are generated and trained, using varying combinations of the features listed above, and it is found that the optimal feature combination is dependent on the stance target. [Lai et al., 2016] manage to outperform all participants in the SemEval-2016 Task 6 competition by a comfortable margin.

Hierarchical Attention Networks (HAN) The model proposed by [Yang et al., 2016] tackles document classification by focusing on document structure. The network holds two continually updated states in memory - one on word-level and one on sentence level. State updates are controlled by a reset gate (r) and an update gate (z). Updates are calculated as follows, with $h_{t\text{-cand}}$ being a candidate state for h_t

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot h_{t-cand}$$

Words are encoded into a matrix using a bi-directional GRU, thus incorporating contextual information in the annotation. Seeing as not all words are found to contribute equally to the classification task, the most influential words are extracted and aggregated to form a sentence vector. Sentences are similarly encoded using a bi-directional GRU, influential sentences are extracted and aggregated to form a document vector. The document vector is finally fed through a softmax layer, and training loss is calculated using the negative log likelihood.

The model is tested on six large-scale document classification datasets, including sentiment and topic classification tasks. As baseline [Yang et al., 2016] implement a 18 models including linear models, SVMs, CNNs an LSMT and a gated RNN. For every one of the six classification tasks, the HAN implementation manages to outperform all baseline models. Furthermore, it is found that non-hierarchical neural network-based models do not outperform traditional methods such as SVM for large-scale document classification tasks.

Bidirectional LSTM [Augenstein et al., 2016] seek to build a model for stance detection for instances where no training data is available for the stance target, and the target is not necessarily present in the tweet. The model is based on two LSTMs - one for stance targets and one for tweets, where the output of the last node in the stance target LSTM is used as input for the first node in the tweet LSTM, making the model conditional on the stance target. A model is trained going left to right over the words in first target then tweet, and another going right to left on first target then tweet. The output vectors of these two models are the basis of the final classification, making it bidirectional. Classification is performed using a non-linear projection such as the hyperbolic tangent using the function below, where W is a weight matrix and h_n is the concatenation of the right-oriented and left-oriented output vectors

$$c = tanh(W * h_n)$$

[Augenstein et al., 2016] train and test this model on the SemEval 2016-Task 6 dataset, managing to rank in the top two with their unsupervised approach. The dataset is then enhanced with automatically labeled tweets, in a technique which resembles those applied by the top two performing teams in SemEval 2016-Task 6. Using this enhanced dataset, [Augenstein et al., 2016] manage to outperform all other entries to the competition, despite using word embeddings as their only features.

3 Relevant work outside NLP classification

In this section, work within the field of machine learning deemed potentially relevant to the thesis project, yet not within the field of NLP classification, is presented.

Multitask Learning [Collobert and Weston, 2008] implement a Time Delayed Neural Network (TDNN), and using a multitask learning approach achieve state of the art results for a number of NLP tasks. At the time, the common approaches to most NLP tasks applied a SVM, requiring a large number of difficult to extract features for more advanced tasks such as semantic role labeling (SRL). Part of the significance of [Collobert and Weston, 2008]'s implementation lies in the fact that it achieves results comparable to and for some tasks exceeding the classic SVMs, using only word embeddings as features, with impressively low run-time. The tasks on which the model is tested are

- Generating POS tags
- Chunking
- Named Entity Recognition
- SRL
- Language model prediction
- Finding semantically related words (synonyms)

[Collobert and Weston, 2008]'s implementation consists of lookup tables, a single convolutional layer, max-over-time pooling and a softmax activation function. Models were trained on all combinations of aforementioned tasks, sharing lookup tables between tasks, to uncover whether a multitask learning approach would improve performance. The top performing combination was found to be SRL and language model prediction, closely followed by the combination of all tasks.

Dataset creation for Machine Learning [Zubiaga et al., 2018] seek to expand the field of rumor identification and rumor stance classification by exploring the impact of a rumor's veracity on its spread, both before and after the veracity has been determined. Furthermore, [Zubiaga et al., 2018] generate a model for automatic collection of rumors, that do not need to already have been defined, as well as the classification and annotation of these. A complex annotation scheme is applied for data collection, visualized below.

Based on the generated dataset, [Zubiaga et al., 2018] find that it generally takes significantly longer for veracity to be determined for false rumors than for true rumors, and that rumors of undetermined veracity tend to be shared more often than rumors of determined

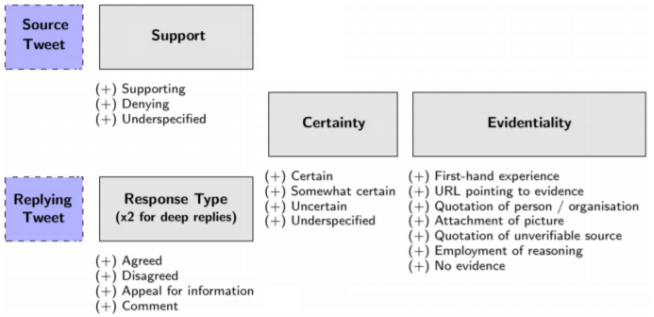

Figure 2: Data annotation scheme, Zubiaga et al. 2018

veracity. Furthermore, it is found that unverified rumors in general tend to garner more support than verified ones, and that users tend to provide more evidence for a given rumor when its veracity is undetermined.

[Ferreira and Vlachos, 2016] enhance an existing dataset of 300 rumors and 2.595 associated news articles, by manually annotating rumors with their veracity and articles with their stance towards the rumor as for, against or observing. Furthermore, articles are summarized into concise headlines more suitable for NLP classification than the full articles. The generated dataset is both applicable for stance classification tasks when using the connections between rumors and articles, as well as for fact checking tasks when only using the rumors labeled for their veracity. The resulting dataset is not significantly skewed towards any class, and the majority class of news articles, for instance, holds 47 % of the datapoints.

4 Provisional Project Plan

The following project plan is provisional, and will possibly be due to change based on the feedback from Leon Derczynski based on this paper, as well as on the insights acquired through the data collection process and implementation of machine learning models. The project will contain two main phases, with a number of tasks connected to each, as well as a number of possible extensions to be implemented if the time schedule allows it.

Dataset Generation The dataset generated for the project will be a number of quotes from Danish politicians currently in the Danish parliament, scraped from news articles. A goal of the dataset generation is to set clear parameters for the scraping task, which can be re-implemented by other researchers, to allow future extension of the dataset with new articles and data sources.

In Figure 4 is an example of a possible quote to be extracted from a news site, related to the topic *indvandring* (immigration) and the Danish politician Martin Henriksen, member of Dansk Folkeparti. The quote reads *"These people do not have that much money, and the more expensive we make the ferry ticket, the more difficult it will be for them to get to the main land"*. In Figure 5 is the headline connected to the article, from which the quote is extracted, reading *"New asylum transfer income can be in violation of the Constitution: "You must be able to live a dignified life""*. Articles of interest might be identified solely based on headlines or on the full article text, possibly using a boolean or regular expression search query.

Figure 3: Example of quote from Altinget.dk

Figure 4: Example of title from Altinget.dk

After extraction, manual cleaning of the dataset will be necessary, labeling articles and quotes identified as false positives for each topic. This will allow the use of the dataset for further research within NLP classification, as a task might be set up to identify relevant and irrelevant news articles for a given subject, from the corpus of labeled data.

Stance Classification The stance classification task will initially be implemented as binary classification of quotes for/against a specific target. It is expected that occurrence of neutral quotes will be very sparse, as it seems unlikely for journalists to include quotes in their articles, which do not have a stance on the topic of the article. A choice has not been made regarding the final model type to be used in the project, but given the very successful results of both [Li et al., 2017] and [Augenstein et al., 2016] in implementing a bidirectional LSTM model, this is a strong contender for the approach.

A feature engineering approach will be applied. Possible features include features extracted directly from the text such as POS tags, n-grams, lexical patterns and word-embeddings, as well as contextual features such as the quoted politician's party affiliation, majority class party stance for the topic and majority class politician stance for the topic. Use of features such as quote sentiment and use of positive and negative words rely on which available resources are found for Danish.

Possible extensions A classification model which can identify true/false positives during data scraping would be a logical extension of the project, making it easier for other researchers to expand the dataset. Such a model would furthermore make it possible to extend the project with experiments using multi-task learning between the true/false positive classification model and the stance classification model, comparing results with the standalone model implementations.

4.1 Project timetables

Below two possible project timetables are presented. The first covers the scenario where no extensions are implemented, and focus is put entirely on the core task of stance classification. The second includes implementation of a classifier for data scraping and implementation of multi-task learning across the two classifiers. Initially the goal will be to adhere to the second timetable including the two extensions. In the case of unforeseen challenges that result in pushed deadlines, the first timetable will be used instead.

Table 2: Timetable focusing on core functionality

Agreements	**Date**
Start date	01.02.19
Feedback on disposition and theoretical approaches	Week 7
Discussion of methodical questions	Week 8
Collecting empirical data	11.02.19 - 04.03.19
Annotation of empirical data	04.03.19 - 08.04.19
Feedback on theoretical chapters	Week 12
Discussion of analytical approach	Week 13
Implementation of stance detection model	25.03.19 - 29.04.19
Feedback on analysis	Week 18
Results and error analysis	29.04.19 - 20.05.19
Feedback on discussion and last minute questions	Week 21
Deadline for submission	03.06.19

Table 3: Timetable including both core functionality and extensions

Agreements	**Date**
Start date	01.02.19
Feedback on disposition and theoretical approaches	Week 7
Discussion of methodical questions	Week 8
Collecting empirical data	11.02.19 - 25.02.19
Annotation of empirical data	25.02.19 - 18.03.19
Feedback on theoretical chapters	Week 10
Discussion of analytical approach	Week 11
Implementation of stance detection model	11.03.19 - 01.04.19
Feedback on analysis	Week 15
Results and error analysis	01.04.19 - 22.04.19
Feedback on results and error analysis	Week 17
Implementation of classifier for data scraping	22.04.19 - 06.05.19
Feedback on analysis	Week 19
Implementation of multi-task learning	06.05.19 - 20.05.19
Results and error analysis	20.05.19 - 29.05.19
Feedback on results and error analysis, discussion and last minute questions	Week 22
Deadline for submission	03.06.19

5 Conclusion

Correctly identifying politicians' opinions is a vital task in helping individuals make sense of the political landscape, allowing them to make informed decisions when voting. A number of machine learning approaches to classification of natural language was presented, that could be considered in building a solution to such a task, including SVMs, CNNs, RNNs, LSTMs, Bayes Classifiers, Kernel Classifiers as well as a number of novel approaches. Furthermore, approaches to dataset generation, multi-task learning and feature engineering for NLP were presented.

A project plan was proposed for creating a dataset and building a machine learning-based model which would take on the abovementioned task of classification of politicians' opinions. The proposed approach would identify the task as binary classification for a number of politicians, over a number of political subjects, generating a dataset by scraping quotes from news articles, relying on feature engineering for classification.

References

[Augenstein et al., 2016] Augenstein, I., Rocktäschel, T., Vlachos, A., and Bontcheva, K. (2016). Stance Detection with Bidirectional Conditional Encoding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 876–885, Austin, USA.

[Collins and Duffy, 2001] Collins, M. and Duffy, N. (2001). Convolutional Kernels for Natural Language. In *Proceeding NIPS'01 Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, pages 625–632, Vancouver, Canada.

[Collobert and Weston, 2008] Collobert, R. and Weston, J. (2008). "A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of the 25 th International Conference on Machine Learning*, pages 160–167, Helsinki, Finland.

[Enayet and El-Beltagy, 2017] Enayet, O. and El-Beltagy, S. R. (2017). Determining Rumour and Veracity Support for Rumours on Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017)*, pages 470–474, Vancouver, Canada.

[Ferreira and Vlachos, 2016] Ferreira, W. and Vlachos, A. (2016). Emergent: a novel data-set for stance classification. In *Proceedings of NAACL-HLT 2016*, pages 1163–1168, San Diego, USA.

[Iyyer et al., 2014] Iyyer, M., Enns, P., Boyd-Graber, J., and Resnik, P. (2014). Political Ideology Detection Using Recursive Neural Networks. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1113–1122, Baltimore, Maryland, USA.

[Johnson and Goldwasser, 2016] Johnson, K. and Goldwasser, D. (2016). "All I know about politics is what I read in Twitter": Weakly Supervised Models for Extracting Politicians' Stances From Twitter. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2966–2977, Osaka, Japan.

[Kim, 2014] Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar.

[Kochina et al., 2017] Kochina, E., Liakata, M., and Augenstein, I. (2017). Turing at SemEval-2017 Task 8: Sequential Approach to Rumour Stance Classification with Branch-LSTM. In *Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017)*, pages 475–480, Vancouver, Canada.

[Lai et al., 2016] Lai, M., Farias, D. I. H., Patti, V., and Rosso, P. (2016). Friends and Enemies of Clinton and Trump: Using Context for Detecting Stance in Political Tweets. In *Advances in Computational Intelligence: 15th Mexican International Conference on Artificial Intelligence, MICAI 2016*, pages 155–168, Cancun, Mexico.

[Li et al., 2017] Li, X., Chen, W., Wang, T., and Huang, W. (2017). Target-Specific Convolutional Bidirectional LSTM Neural Network for Political Ideology Analysis. In *Web and Big Data. APWeb-WAIM 2017.*, pages 64–72, Beijing, China.

[Ma et al., 2018a] Ma, J., Gao, W., and Wong, K.-F. (2018a). Detect Rumor and Stance Jointly by Neural Multi-task Learning. In *Proceeding WWW '18 Companion Proceedings of The Web Conference 2018*, pages 585–593, Lyon, France.

[Ma et al., 2018b] Ma, J., Gao, W., and Wong, K.-F. (2018b). Detection of stance and sentiment modifiers in political blogs. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers)*, pages 1980–1989, Melbourne, Australia.

[Mohammad et al., 2017] Mohammad, S. M., Sobhani, P., and Kiritchenko, S. (2017). Stance and sentiment in tweets. *ACM Transactions on Internet Technology (TOIT) - Special Issue on Argumentation in Social Media and Regular Papers.*

[Qazvinian et al., 2011] Qazvinian, V., Rosengren, E., Radev, D. R., and Mei, Q. (2011). Rumor has it: identifying misinformation in microblogs. In *EMNLP '11 Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1589–1599, Edinburgh, United Kingdom.

[Skeppstedt et al., 2017] Skeppstedt, M., Simaki, V., Paradis, C., and Kerren, A. (2017). Detection of stance and sentiment modifiers in political blogs. In *19th International Conference, SPECOM 2017*, pages 1589–1599, Hatfield, United Kingdom.

[Wiebe et al., 2004] Wiebe, J., Wilson, T., Bruce, R., Bell, M., and Martin, M. (2004). Learning subjective language. *Computational Linguistics, Volume 30 Issue 3.*

[Yang et al., 2016] Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., and Hovy, E. (2016). Hierarchical Attention Networks for Document Classification. In *Proceedings of NAACL-HLT*, pages 1480–1489, San Diego, USA.

[Zeng et al., 2016] Zeng, L., Starbird, K., and Spiro, E. S. (2016). Unconfirmed: Classifying Rumor Stance in Crisis-Related Social Media Messages. In *Proceedings of the Tenth International AAAI Conference on Web and Social Media (ICWSM 2016)*, pages 747–750, Cologne, Germany.

[Zubiaga et al., 2018] Zubiaga, A., Liakata, M., Procter, R., Hoi, G. W. S., and Tolmie, P. (2018). Analysing how people orient to and spread rumours in social media by looking at conversational threads. *ACM Computing Surveys, Volume 52 Issue 2*.