Classifying COVID-19 News on Sina Weibo

Arunachal Sen

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science

University of Washington

2021

Reading Committee:

Gina-Anne Levow, Chair

Fei Xia

Program Authorized to Offer Degree:
Department of Linguistics

University of Washington

Abstract

Classifying COVID-19 News on Sina Weibo

Arunachal Sen

Chair of the Supervisory Committee:
Professor Gina-Anne Levow
Linguistics

This thesis addresses the classification of Sina Weibo news related to the COVID-19 pandemic by using sentiment analysis. The design is a comparison study, involving four different systems. The systems were chosen after extensive reading on different approaches to sentiment analysis in Mandarin Chinese. These systems are neural network, k-medoids, HMM, and SVM. The core work of this thesis was in fine-tuning these systems to work with a small dataset of less than 3,000 examples. The final results from numerous experiments showed that the SVM and HMM systems achieved the highest results, followed by the neural network and k-medoids systems. Findings of this study showed that keyword frequency within a news category is not necessarily sufficient to ensure correct classification. Also, posts with names and satire remain challenging to classify and could be investigated further.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1
INTRODUCTION

While fake news is not a new problem, online channels have allowed for easier, faster, and more widespread dissemination of deceptive and misleading rumors than was previously possible. Fake news often appears in conjunction with major events such as elections, terrorist attacks, and more recently, the ongoing COVID-19 pandemic.

1.1 Current Methods of Addressing Fake News

Some current systems for addressing fake news involve manual flagging and review, which can be slow, giving fake news more time to proliferate. While postings can also be censored based on certain keywords, this approach may also block some legitimate posts. From the standpoint of Natural Language Processing (NLP), automated detection of fake news is basically a classification task, but it is far from easy. A comprehensive automated solution might well involve a multi-modal approach that analyzes photos, videos, text, and user account attributes associated with the post, as noted by Ajao et al. (2019). The scope of this thesis is limited to linguistic factors, specifically those within the text of the posting. Also, this study does not seek to define what constitutes fake news, which is a potentially controversial topic. Rather, this study considers pre-labeled examples to be the source of truth; it then classifies news using a popular NLP tool, sentiment analysis.

1.2 Research on Sentiment Analysis in Different Languages

Sentiment analysis research in English has been ongoing for several decades and has an abundance of lexical resources along with various algorithms which can be leveraged for the task. Sentiment analysis research in some other languages has not received as much

attention, however. This gives an opportunity for researchers to address the research gaps by developing new sentiment analysis resources and tuning existing systems to work well with a given language. This study focuses on Chinese (Mandarin) social media posts on Sina Weibo, specifically those related to the COVID-19 pandemic. Chapter 3 gives more details on the dataset. For an overview of literature dealing with both Chinese sentiment analysis and fake news sentiment analysis, please see Chapter 2.

1.3 Overview of the Study

The study will compare the accuracy of four different sentiment analysis systems at classifying news on social media. The following systems will be implemented:

1. **FlairNLP**, hereafter referred to as the Neural Network system

   FlairNLP is an open-source NLP system. The classification module is based on Pytorch, a Deep Neural Network (DNN) package. FlairNLP offers the ability for researchers to use customized word embeddings along with ones already built into the system. Given the record of FlairNLP for high performance, the goal here is to find the word embeddings that give the highest accuracy on the new dataset.

2. **Hidden Markov Model**, hereafter referred to as the HMM system

   The HMM is trained to give the probability of fake or real news by using different word features for the hidden states. The goal is to find word features that best predict news categories.

3. **K-medoids**, hereafter referred to as the k-medoids system

   Emotions are tagged according to a lexical resource. Then, the longest common sequence of emotions between two posts is used to cluster the posts. One goal of this method is to see whether fake/real news categories actually form separate clusters.

4. **Support Vector Machine**, hereafter referred to as the SVM system

   This approach considers multiple types of emotions and how they relate to categories of news. Sentiment from the sentence level is strung together to form sequences, from which rules are mined. The goal is to see whether these class sequential rules can accurately predict real/fake news alongside more traditional word and emotion intensity features.

Algorithms used in these four systems are discussed in Chapter 4. The input to each system is text from a preprocessing module. This module cleans the social media text and then segments it. The main module of each system loads this input and then instantiates, trains, and tests the model. Finally, the results from the test are evaluated. More details on the setup of each system are provided in Chapter 3. The experiments and final results are covered in Chapter 5, while Chapter 6 deals with the error analysis.

Chapter 2

LITERATURE REVIEW

While Chinese sentiment analysis is less researched than that of English, a number of studies have appeared within the last decade, indicating a growing interest among researchers. Peng et al. (2017) give a comprehensive overview regarding Chinese sentiment analysis. Their classification of studies by type serves as a rough outline for this literature review. Approaches to Chinese sentiment analysis can be categorized as monolingual or bilingual. Monolingual approaches directly analyze sentiment from Chinese text using Chinese-based resources, while bilingual approaches use machine translation and English-based resources for sentiment analysis on the original Chinese text.

The number of studies addressing sentiment analysis for fake news detection is also low. However, existing studies are generally fairly recent. A number of studies on fake news use sentiment analysis along with other methods both within and outside the scope of computational linguistics. Ajao et al. (2019) found that utilizing sentiment analysis for fake news detection gives better results than methods that do not consider sentiment. However, Cui et al. (2019) note that even with recent developments in the field, early detection of fake news remains a difficult problem. In the following sections, Chinese sentiment analysis studies and fake news studies are discussed together according to the methods used. Unfortunately, there is no easy way to compare studies. Results do not consistently include accuracy and F1 scores. Also, the datasets are significantly different between studies. Thus, this literature review focuses on what parts of a given system worked reasonably well, and which studies have a domain similar to that of this present research.

2.1 Machine Learning Approaches

Neural network approaches have become increasingly popular recently. However, non-neural network approaches such as Naïve Bayes, Maximum Entropy, and Support Vector Machine (SVM) still remain relevant as evidenced in studies within the past decade. Resources that support machine learning include corpora. One well-known corpus is NTU-Multilingual Corpus (Tan and Bond, 2011), which contains 375,000 words in six languages, including English and Chinese. Machine learning constitutes a major category of approaches to sentiment analysis and can be divided into two subcategories based on the emphasis of the study, namely, feature engineering for a classifier or the classification method itself. The following sections discuss these two subcategories in more depth.

2.1.1 Feature Engineering

Research into feature-based approaches includes some well-known features commonly used in English sentiment analysis, as well as features unique to Chinese. In an earlier study, Tan and Zhang (2008) used the information gain (IG), CHI square (CHI), document frequency (DF), and mutual information (MI) calculated over the terms resulting from POS parsing and tagging the document. The following classifier systems were compared: centroid, k-nearest neighbor, winnow, Naïve Bayes, and SVM. They found that using IG as a feature with an SVM classifier gave the best performance. However, the results were heavily domain-dependent. The SVM classifier and features above are part of the SVM system in this study, which is discussed more in Chapter 3.

Zhai et al. (2011) investigated substring, substring-group, key-substring group, and sentiment-word features, which previously had not been extensively researched. Substrings are commonly used in NLP and include n-gram features. Substring-group refers to clustering a larger number of substring features into a smaller, more manageable number of equivalence groups using suffix tree techniques. The suffix tree starts with short sentences which result from splitting on punctuation; smaller substrings are then added to the tree, representing features.

The set of matching features is the substring group. If the number of features is still too large, features that meet certain criteria can be selected to form a key (feature)-substring group. Sentiment word features are based on listings in a sentiment lexicon. This study found character bigrams to be superior to other n-grams. However, there is greater performance improvement potential from substring-group features. Also, they discovered varying-length features were generally better than fixed-length features. The features in their study almost always outperformed the baseline lexicon-based approach.

In a more recent study, Su et al. (2014) extracted semantic features from the text using lexicon-based and part-of-speech based feature selection. Then they used Word2Vec with a neural network model to learn the vector representation of the selected features. Classification done with SVM using these features as vectors achieved very good accuracy. Although the results were fairly good, dealing with high dimensional feature vectors in SVM posed a significant problem. Also, the two methods of feature selection were considered inadequate for fully capturing the sentiment of each sentence.

One drawback of using solely the common features from English sentiment analysis is that nuances of Chinese might be lost. Several Chinese-specific word embeddings have been developed to address this problem. Character-enhanced Word Embedding (CWE) created by Chen et al. (2015a) is a word embedding based on a continuous bag of words model (CBOW) which attempts to capture the internal information of a word at the character level, unlike most embeddings, which focus on the word level.

The Joint learning Word Embeddings (JWE) developed by (Yu et al., 2017) goes even further by creating a joint embedding of Chinese words, characters, and subcharacter components. This embedding is also based on the CBOW model. Both CWE and JWE give improvements over the baselines for word similarity computation and word analogy reasoning. Another CBOW-based embedding is the Radical-Enhanced Chinese Word Embedding (RECWE) of Chen and Hu (2018). Here, radicals have been converted to the character they represent, which captures semantic information better than using an unconverted radical. Glyph-Enhanced Word Embeddings (GWE) from Su and Lee (2017) involve context char-

acter glyph features. They found that learning a word representation from a glyph features sequence does not work well, however, the features can be used as an enhancement to an existing word embedding such as CWE.

Finally, Cao et al. (2018) introduced Chinese word2vec (cw2vec), a word embedding based on character stroke n-grams. Testing results showed this embedding outperformed not only word2vec and GloVe, but also CWE, JWE, and GWE on tasks such as named entity recognition, word similarity, word analogy, and text classification, which is the goal of this present research. These five word embeddings, trained on a Weibo corpus, are used to create a text classifier in the Neural Network system, which is discussed more in Chapter 3.

2.1.2 Classification Methods

Based on the amount of literature, classification methods themselves appear to have received more attention from researchers than feature engineering. Typically, a single classification algorithm is deployed, however, a few studies have included ensemble methods, which harness multiple learning algorithms.

Ensemble methods involve a combination of algorithms working together. One example of ensemble methods is Multiple Probabilistic Reasoning Model which is based on random feature space division (Xu et al., 2013). Their study overcame the problems of highly sparse features and hidden sentiment in short text of micro movie reviews. Principal Component Analysis was used to compact the feature space, which was subsequently divided into multiple subspaces. These multiple subspaces allow for the use of fine-grained sentiment. However, one drawback is the long training time required for the algorithm. In the domain of fake news detection, Ahmad et al. (2020) also used ensemble methods to classify fake news articles from various domains. They demonstrated that a combination of learning algorithms, namely, random forest, voting, bagging, and boosting classifiers performed better than a single learning algorithm.

Neural networks have become an increasingly popular approach to many computational linguistics tasks, and sentiment analysis is no exception. In one of the earlier neural network

studies involving Chinese sentiment analysis, Li et al. (2014) created a Chinese Sentiment Treebank of movie reviews, and then used deep learning in their Recursive Neural Deep Model for classifying the sentiment as positive or negative. Their classifier worked at the sentence level and outperformed conventional models of the time by a good margin.

A multi-channel convolutional neural network was implemented by Liu et al. (2018). The purpose of multiple channels was to achieve multiple representations of Chinese–word, character, and pinyin. While the individual strengths of these representations differed, they appeared to complement each other well when used in combination. This model achieved better performance than conventional SVM models with n-gram features.

Even more recently, Kula et al. (2020) achieved quite good results for fake news detection by using FlairNLP, an NLP package based on neural networks. Accuracy was significantly different depending on the word embedding used. Notwithstanding this remarkable accuracy, the authors noted that fake news contains multiple subcategories which would need to be addressed in a new model. FlairNLP was selected as the primary component of the Neural Network System for this study.

While neural networks are undeniably popular, other classification methods are still featured in numerous studies. Bhutani et al. (2019) hypothesized that news sentiment is an important deciding factor in news classification. For example, fake news often has negative sentiment (i.e. false allegations). They used Term Frequency-Inverse Document Frequency (TF-IDF), sentiment, and cosine similarity values as features when training fake news detection models for Twitter posts. The Naïve Bayes and Random Forest algorithms were used for classification, with the latter achieving better results.

For Weibo posts, Liu et al. (2015) used a Hidden Markov Model (HMM) based on text features of words. Particle Swarm Optimization (PSO) was used to optimize the sequence of hidden states. Results showed a better accuracy for happiness and fear, but only fair accuracy for anger, surprise, and sadness. The HMM system implemented in this study roughly follows their design. The setup and technical workings are discussed in Chapters 3 and 4, respectively.

Chen et al. (2015b) separated subjectivity detection and polarity classification into two isolated stages to avoid error propagation common in a pipeline approach. They then used Markov logic to learn the different feature sets of the two stages. Overall performance was decent for subjectivity and somewhat lower for polarity. However, performance for objective and neutral polarity classification was low, showing an area for future improvement.

Liu and Chen (2015) depart from the traditional binary or multiclass sentiment labeling classification with a multi-label analysis, while also comparing multiple classification methods–Productive Clustering Trees (PCT), SVM, and k-nearest neighbor (kNN). Three different sentiment dictionaries, namely, Dalian University of Technology Sentiment Dictionary (DUTIR), HowNet Dictionary, and National Taiwan University Sentiment Dictionary (NTUSD) were used. DUTIR, HowNet, and an augmented version of NTUSD were selected for use in this study. However, none of the dictionaries were customized to the datasets.

Finally, Wei et al. (2014) used k-medoids clustering algorithm which does not require training and works well on short text such as Weibo microblog posts. The text was converted into sequences of emotion based on a sentiment dictionary. Classification in this model is done by finding the shortest distance between the medoids and unclassified posts, given the longest common subsequence of emotions between them. While this method is fairly straightforward, it depends heavily on the accuracy of the sentiment dictionary. A k-medoids system was implemented in this study; further details are covered in Chapter 3.

2.2 Knowledge-based Approaches

Knowledge-based approaches usually involve resources such as lexicons. There do not appear to be many studies that use a purely lexical approach; rather researchers favor a syntax-based approach that contains rules and logic. The syntax-based approach is more appealing because it offers flexibility when designing the rules compared to a more rigid strictly lexicon-based approach. Regarding the need for more development in this area, Cambria and Hussain (2015) call for more research into concept-level reasoning. They argue that concept-level reasoning more accurately represents real-world truth compared to other approaches.

2.2.1 Lexical Resources

Lexicons can be used for knowledge-based approaches, or to augment machine learning based approaches. A lexicon contains a list of words and associated sentiment. Lexicons can be manually constructed based on an existing dictionary, or constructed computationally using a corpus. The manual approach may yield a higher quality product, but requires a significant amount of time and human resources. An example of an existing dictionary is HowNet, developed by Dong et al. (2010). However, there do not appear to have been any further updates to HowNet since 2015. Liu et al. (2013) combined a domain-specific lexicon with HowNet for sentiment analysis, addressing the problem of domain adaptability. Xu et al. (2010) took another approach, augmenting existing dictionaries with unlabeled corpus data. Their resulting dictionary allows for the computation of polarity strength. Finally, Peng and Cambria (2017) present CSenticNet, a system that leverages the English-based SenticNet system and the Chinese NTU Multilingual Corpus. SenticNet is a hybrid sentiment analysis system that uses both linguistic and knowledge bases and machine learning.

2.2.2 Rules and Logic

In the research of fake news, Cui et al. (2019) employed Valance Aware Dictionary for Sentiment Reasoning (VADER), a rule and lexicon-based resource, to classify sentiment expressed on social media. They found that fake news tends to exhibit stronger negative or positive polarity than real news. In other words, fake news is less neutral than real news.

Lexical resources provide a rich variety of attributes that can give clues regarding sentiment. Zhang et al. (2009) present a straightforward rule-based method where word dependency is used to determine the sentiment of a sentence. This sentiment is then aggregated to obtain the document sentiment. This approach achieves fairly good accuracy over multiple datasets. One benefit of such an approach is that it eliminates the need for manual labeling of training data. Rule creation through learning was suggested as an area for future work. A similar approach to sentiment aggregation is used in the SVM system of this study.

Another study featuring the rule-based approach is Xiong et al. (2014). First, appraisers, degree adverbs, and negations (ADN scoring) are combined into rules. Word distances serve as rule constraints, along with strengths of degree adverbs and appraisers. PSO is used to optimize the constraint thresholds and strengths for the rules. The model is also extendable to other domains such as microblog.

Positive and negative sentiment can also be depicted as a continuum as do Zagibalov and Carroll (2008) They mined document opinions using unsupervised techniques such as one-word seed vocabulary, the criterion of sentiment density, and iterative retraining of sentiment during processing. While the system is fairly simple and easily portable to other domains, the accuracy relies very heavily on the selection of a good initial seed vocabulary.

Yet another approach is treating Chinese opinion elements as a combination problem where topic, feature, item, and opinion words are combined to predict an opinion (Wu and Chiang, 2015). While fairly good results were obtained on a dataset of forum review messages, the authors mentioned a need for improvement in the analysis of narrative and hypothetical sentences.

In Quan et al. (2013), an integrated approach involving a sentiment lexicon and dependency parsing for sentiment classification was used. Evaluation objects and sentiment were extracted based on dependency, and Hownet was used to find similarity between words. Weights were allocated and finally a positive or negative classification was realized. Results are also fairly good, but there is room for improvement in subjective vs. objective classification.

2.3 Mixed-Model Approaches

The mixed-model approach represents a fusion of machine learning and knowledge-based methods. Zhang and He (2013) used an Enhanced Self-Training (EST) algorithm to leverage both a lexicon and a corpus with an agreement strategy for choosing optimal labeled data. Two partitions were split from a test dataset. Classification results were then combined into training (pseudo-labeled) and test (unlabeled) sets. The classifier was initially trained with

the pseudo-labeled data and then entered another self-learning cycle. Notwithstanding the high accuracy, the authors believe self-labeling could be further improved by considering linguistic knowledge while creating the test set.

In a comparison study, Yuan et al. (2013) classified microblog sentiment using both supervised machine learning and unsupervised knowledge-based approaches. The supervised approaches were Naïve Bayes, Maximum Entropy, and Random Forest algorithms, while the unsupervised approach consisted of multiple lexicons and a simple count of the sentiment instances. Machine learning achieved the highest accuracy, while knowledge-based accuracy was significantly lower. Similarly, this study is also a comparison study of four methods.

Finally, Wen and Wan (2014) used class sequential rules in their approach. Mixed methods (SVM and lexical) were used for initial emotion labeling of the words in a microblog text. Then, rules were derived from the sequences of emotions. Features were generated using the rules, which formed a basis for an SVM classifier. Notwithstanding this unique approach, the system did not achieve very good results, showing a need for further improvements. Given the opportunity to improve results in the microblog domain, this approach was chosen as the basis for the SVM system in this study.

2.4 Multilingual Approaches

Due to the lack of suitable Chinese lexical resources, some researchers have attempted to use English-based resources for Chinese sentiment analysis. Wan (2008) translated Chinese reviews into English, which were then analyzed using a knowledge-based approach with a dictionary. The highest accuracy came from a combination of Chinese lexicons along with Google and Yahoo translation resources. However, one problem with this approach is the reliance on machine translation, which leaves room for translation errors.

Wei and Pal (2010) proposed using only key parts of translations that are more reliable, instead of the whole translation. They used structured correspondence learning (SCL) to circumvent translation errors by linking the languages at a feature level and retained only key pivot features. However, the average accuracy of the best model did not improve greatly.

He et al. (2010) used a Latent Dirichlet Allocation (LDA) model to train on information from a translated English corpus. The highest accuracy is fair, with a gap between target and source languages remaining a significant issue. Interestingly, translating the English lexicons into Chinese and classifying them on a Chinese corpus outperforms translating the corpus from Chinese to English in order to use English lexicons.

2.5 Summary

This literature review shows that a majority of studies on Chinese sentiment analysis are monolingual. They are either machine learning based, knowledge-based, or a combination of the two. This comprehensive review of associated literature helped lay the foundation for the design of this study, which is presented in the following chapter.

Chapter 3

METHODOLOGY

This chapter outlines processes for gathering the data, cleaning the data, running the experiments, and evaluating the results. The studies cited in the literature review, in particular, those dealing with microblogs, were carefully analyzed. Then, four different approaches were selected as the basis for a comparative study.

3.1 Dataset

The initial dataset and accompanying Weibo scraping programs were provided by Yang et al. (2021) at `https://github.com/cyang03/CHECKED`. Fake news posts are from ordinary Weibo users. These posts were subsequently flagged, reviewed by a moderator, and determined to be fake news. Real news posts are those posted on Weibo by an official media outlet, *People's Daily*. An extracted post and its metadata are represented as comma-separated values in this dataset. In this study, only the category label (fake, real) and the text columns were selected for pre-processing.

The initial dataset included posts extracted from December 2019 to August 2020. The posts were selected based on specific keywords related to COVID-19 (Appendix A). After some minor code modifications to allow the scripts to log in to Weibo and account for website design changes since the initial release, the scraper programs were run to extract newer posts from August 2020 to May 2021, creating a final dataset of fake and real news posts which spans 18 months, December 2019-May 2021.

An examination of the dataset showed that fake news posts sometimes contained various emoticons, while real news posts did not. Given the scope of this study is strictly text-based sentiment, it was decided to filter out non-Chinese characters with the exception of

punctuation. This was accomplished using a simple regular expression for the Unicode range representing Chinese, `4E00-9FFF`. The filtered text was then segmented using PyNLPIR, a Python wrapper around the NLPIR/ICTCLAS Chinese segmentation and POS tagging software. The software is available at `https://github.com/tsroten/pynlpir`. In total, there are 2529 examples of real news and 394 examples of fake news. After the order was randomized, data were split into training, development, and testing sets using an 80/10/10 split.

3.2 Resources and Setup

Much of the code in the Neural Network, HMM, and SVM systems comes from existing packages. While the Neural Network system was based on an NLP library with a sentiment analysis component, the HMM and SVM systems use general data modeling and optimization packages and thus required additional sentiment analysis specific coding by the author. The k-medoids system was entirely coded by the author. Given the research challenges specific to this study, namely having fewer Chinese resources and dealing with a rather small dataset of just under 3000 examples, the focus is on tuning the various systems.

3.2.1 Neural Network System

This system utilizes FlairNLP, developed at Humboldt University, Berlin. It can be downloaded at `https://github.com/flairNLP/flair` (Akbik et al., 2019). Training FlairNLP's neural network language models on a CPU is extremely slow. Thus, this system was implemented in a Google Colab notebook, a popular cloud-computing resource among researchers. Most importantly, it includes a GPU which allows for significantly faster training times (`https://colab.research.google.com`). Inspiration for this approach comes from Kula et al. (2020), who published one of the first studies on FlairNLP for fake news detection.

The process of using FlairNLP is very straightforward. FlairNLP has multiple classification models, including `TextClassifier` and `TARSClassifier`. `TextClassifier` uses pre-trained and custom word embeddings, while `TARSClassifier` trains off of a corpus di-

rectly. Both of these models are used in this study. Technical aspects of these approaches are discussed further in Chapter 4. For both models, the train, development, and test data files were loaded into a `ClassificationCorpus` data object. `TARSClassifier` was by far the simplest to set up, while `TextClassifier` required additional steps.

First, the word embeddings were instantiated. FlairNLP has the following pre-trained Chinese embeddings: fastText `WordEmbeddings`, BERT `TransformerWordEmbeddings`, and byte pair `BytePairEmbeddings`. FastText embeddings have been trained on Chinese Wikipedia and Common Crawl web data. BERT and BytePair embeddings have been trained on Chinese Wikipedia. FlairNLP allows embeddings to be stacked together for a combined approach. Custom embeddings were also loaded at this time as `WordEmbeddings`. Provided the embeddings are in Gensim format, detailed at `https://radimrehurek.com/gensim/`, they can be loaded into FlairNLP. Custom embeddings are considered to be those which are not native to FlairNLP. They can be categorized as pre-trained or trained.

Custom pre-trained word embeddings are based on the work of Li et al. (2018); they are skip-gram word vectors from `https://github.com/Embedding/Chinese-Word-Vectors`. The word vectors represent a variety of online and offline domains: Baidu Encyclopedia, Chinese Wikipedia, People's Daily News, ZhiHu Q&A, Weibo, and Chinese Literature. Additional custom embeddings were trained locally on a Weibo corpus provided by the Natural Language Processing and Information Retrieval Sharing Platform (NLPIR), available at `http://www.nlpir.org/wordpress/download/weibo.7z`. The Weibo corpus was selected for the training of word embeddings because the COVID-19 news dataset is also from Weibo. After postprocessing according to the same standards used for the news dataset, the corpus contained 4.9 million posts. Five different types of custom word embeddings were trained: Radical Enhanced Chinese Word Embeddings (RECWE), Glyph-Enhanced Word Embedding (GWE), Joint learning Word Embedding (JWE), Chinese word2vec(cw2vec), and Character-enhanced Word Embedding (CWE), which were discussed in Chapter 2. After training, the embeddings were converted to Gensim format. These custom vectors constitute a new addition made in this study.

Next, the document embeddings were instantiated as `DocumentRNNEmbeddings`, based on the previously described word embeddings. Here, a number of parameters are available such as `hidden_layer_size`, `reproject_words`, and `reproject_words_dimension`. Tuning these settings was part of the experiments, described in Chapter 5. Depending on the approach, either a `TextClassifier` or a `TARSClassifier` was instantiated based on a label dictionary derived from the corpus. Next, a `ModelTrainer` was instantiated using the corpus and the previously created `TextClassifier`, and training commenced. There are several hyper-parameters available; the settings used for various experiments are discussed in more detail in Chapter 5. Code snippets showing the final FlairNLP configuration for training and testing with hyper-parameters are shown in Appendix E. A comparison of the two types of training processes is shown in Figure 1.

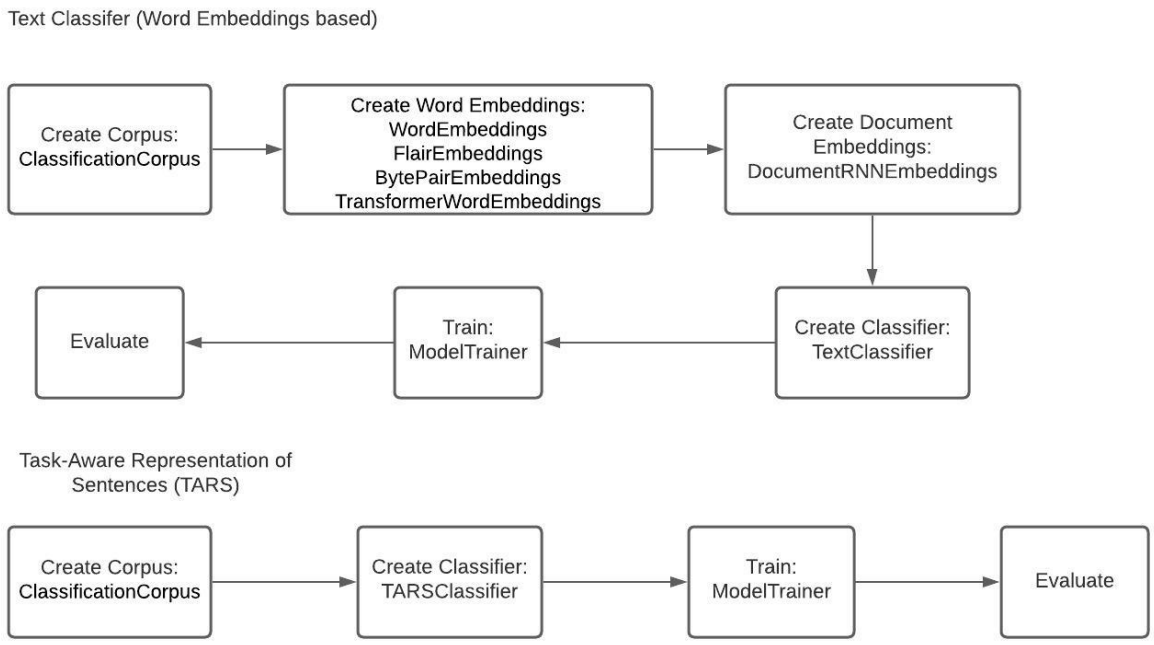

Figure 3.1: FlairNLP Pipelines

3.2.2 HMM System

For this system, two HMMs were constructed–one for real news, and one for fake news. When the models were tested, the category of the model giving the test post the highest probability was chosen as the system prediction. This system is based on the approach described by Liu et al. (2015), where Mutual Information (MI), Chi-Square (CHI), Term-Frequency-Inverse Document Frequency (TF-IDF), and Expected Cross-Entropy (ECE) served as the features for observed variables in the HMM. Information gain (IG) was added in this study as an additional feature.

MI is a quantification of information about one random variable derived from observation of the other. Here it represents the mutual dependence between independent variables term t and news category c_i, and is calculated as follows:

$$MI(t, c_i) = log\frac{P(t|c_i)}{P(t)} \tag{3.1}$$

CHI-square is used for checking lack of independence between the term t and news category c_i, and is calculated as follows:

$$\chi^2(t, c_i) = \frac{(AD - BC)^2 \times N}{(A + B)(C + D)(A + C)(B + D)} \tag{3.2}$$

The letters here represent counts as follows: A is the instances of t in c_i, B is the instances of t not in c_i, C is the instances of non-t that are in c_i, D is the instances of non-t not in c_i, and N is the total number of posts. Higher CHI values suggest more dependence between t and c_i.

TF-IDF represents the ability of term t to distinguish between c_i due to its presence or absence in posts of a specific category. It is calculated in two parts as follows:

$$TF_{i,t} = \frac{N_{i,t}}{\Sigma_k N_{k,t}}; \quad IDF_t = log\left(\frac{N}{n_t} + 0.01\right) \tag{3.3}$$

Here, *TF* indicates the frequency of a term in a given category. $N_{i,t}$ is the number of times a term appears in c_i. *IDF* indicates how much information is provided over the whole dataset,

with n_t being the number of posts containing term t and N being the number of posts in the dataset. The two measures are then combined:

$$TF\text{-}IDF(t, c_i) = TF_{i,t} \times IDF_t \tag{3.4}$$

Expected cross-entropy expresses distances between probability distributions for post category over the news categories, given the term t. It is calculated as follows:

$$ECE(t, c_i) = P(c_i|t) log \frac{P(c_i|t)}{P(c_i)} \tag{3.5}$$

IG indicates the entropy reduction achieved by the presence of term t in a post. Entropy of the text with and without the t is calculated over all categories C in this study. Then the difference is taken as follows:

$$IG(t) = -\sum_{i=1}^{|C|} P(C_i) log P(C_i) + P(t) \sum_{i=1}^{|C|} P(C_i|t) log P(C_i|t) + P(\bar{t}) \sum_{i=1}^{|C|} P(C_i|\bar{t}) log P(C_i|\bar{t}) \tag{3.6}$$

Features were calculated for each word of a post. The values of the features were then summed together for all the words of a post, and the average was taken. The resulting values formed a feature vector. The features were then used to train the HMMs. For this system, the HMMLearn package (`https://github.com/hmmlearn/hmmlearn`) was used to implement the models. The features for a given model were all given weights ranging from 0 to 1. These weights were determined using Particle Swarm Optimization (PSO). More technical details of the HMMs and optimization are given in Chapter 4. Code snippets showing the final model configuration and PSO tuning hyper-parameters are given in Appendix E.

The model was then initially tested; posts that got the highest probability under a respective model were considered to be associated with the category represented by that model. Development data were then used to tune the model. The probabilities of each feature vector T under categories c_i represented by both models were used to calculate entropy:

$$\phi(T) = -\sum_i P(c_i|T) log(c_i|T) \tag{3.7}$$

Increasingly negative values of entropy indicate potentially more discriminative posts. Development data were added to the training data if it met an entropy threshold of -10. An iterative process of training and tuning was used to find optimal entropy settings, shown in Figure 2.

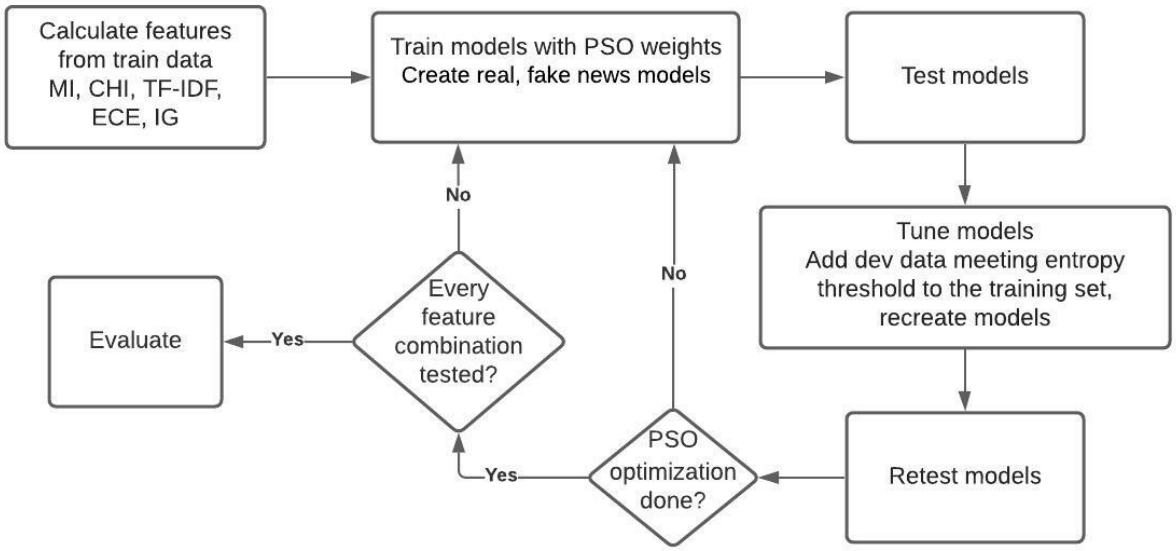

Figure 3.2: HMM Pipeline

3.2.3 K-medoids System

The k-medoids system design is based on the research of Wei et al. (2014). A key difference in the present study is that different sentiment dictionaries were selected for comparison of performance: Augmented National Taiwan University Sentiment Dictionary (ANTUSD), HowNet, and Dalian University of Technology Information Retrieval Lab sentiment dictionary, known as DUTIR. Also, additional features were added, namely word, emotion, and emotion intensity.

Word vectors were created from the segmented text and stored in a sparse vector representation. Creating the other features involved checking the sentiment dictionaries. In ANTUSD, the *machine score* was selected for matching entries. This decimal value can range from -1 to 1 and represents positive, negative, and neutral sentiment. For values less than 0, the sentiment was considered to be negative; for values greater than 0, the sentiment was considered to be positive. In HowNet, scores are not given, but rather the words are categorized according to positive/negative emotion and evaluation. Strings of sentiment values were created for positive and negative words found in the ANTUSD and HowNet by coding the words as 0 (negative) and 1 (positive), respectively. In DUTIR, there are seven emotion categories: anger, disgust, fear, happiness, like, sadness, and surprise. There were also 21 subcategories of emotion, with lexicon entries tied to a subcategory. The subcategories were mapped to the seven main categories, which serve as the basis for the DUTIR-based features. The intensity of emotion is also given as an integer. The emotion categories were coded as strings of integers based on the matching words from the text. DUTIR was also used to create the emotion and emotion intensity vectors which involve simple counts for each emotion and a sum of emotion intensities for each example, respectively. The sentiment strings were then evaluated to find the longest common subsequence (LCS). The LCS does not require the sequence of characters to be consecutive. However, the order of appearance in the strings (i.e. left to right) matters. Non-matching portions of the parent strings are skipped when creating the final sequence.

Next, the model was implemented using custom code written by the author. This was done because results using prepackaged code were not satisfactory. This makes the k-medoids system unique from other systems in that it does not have many tunable hyper-parameters. Rather, tuning largely involved testing initial medoid selection and a variety of distance measures (i.e. cosine) for determining distances between examples in the word, emotion, and emotion intensity vector representations, as detailed in Chapter 5. Code showing an example distance computation is given in Appendix E. Distance for sentiment string similarity representations was based on LCS, shown in the calculation used by Wei et al. (2014):

$$Distance(S1, S2) = 1 - \frac{LCS\ \ Length(S1, S2)}{max(S1, S2)} \tag{3.8}$$

Repeatedly calculating the distances as the model seeks convergence is a bit slow. To solve this problem, the calculations were done once for all possible parent string pairs and stored in a table for reference as the algorithm runs. Training and development data are not needed for creating a k-medoids model. Thus, only the test data were used for consistency when evaluating against other approaches. K-medoids differs from the other systems in that classification of every example using the emotion string representation is not possible if no emotion words in the example match the particular lexicon being tested. In this case, examples without emotion string representation were simply excluded from testing. More technical details on k-medoids model and LCS computation are found in Chapter 4. The overall layout of the system is shown in Figure 3.

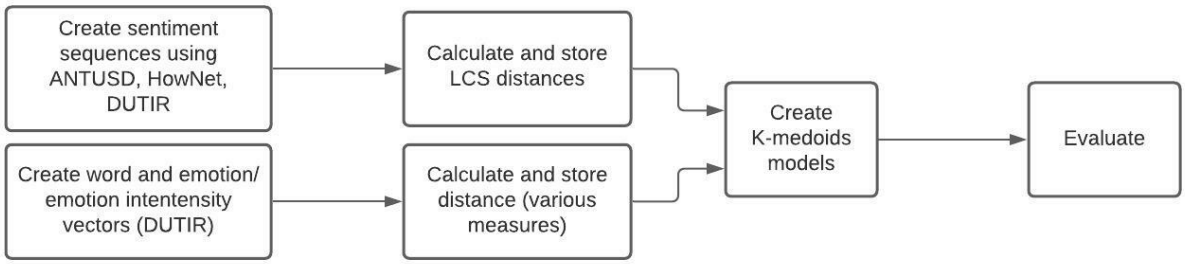

Figure 3.3: K-Medoids Pipeline

3.2.4 SVM System

SVM is a well-known machine learning classifier; the design of the system is based on Wen and Wan (2014). The features for the classifier come from the words, emotions, and class sequential rules, with the rules being the key feature of this approach. One special addition in this study is that different rule structures were tested to determine the most suitable one for the dataset, as discussed in Chapter 5.

Word features are binary, based on the presence of a word in a post. A list of unique words was first created from the training file. Each of those unique words represents an item in a word vector, which is a sparse vector representation. Emotion features and rules are based on the categories of DUTIR. Emotion vectors were created for each example, using a process similar to that from the k-medoids system described previously.

Unlike the process for creating emotion vectors, the rules creation process only captures the strongest emotion for each sentence. Emotions in all categories were aggregated based on matching lexicon keywords in an iterative process for each sentence. When punctuation was encountered, indicating the end of the sentence, the dominant emotion was selected. Conjunctions (based on the original study), adjectives, and adverbs (new to this study) were also selected, based on tagging from PyNLPIR POS tagger. Words selected based on POS were assigned a numeric identifier. Thus, the training data posts were effectively converted into sequences of integers representing emotions and matching POS keywords with the corresponding real/fake classification. A rules file was then generated as input for the mining algorithm.

Top-k Class Sequential Rules algorithm from the SPMF package (`http://www.philippe-fournier-viger.com/spmf/index.php`) was then used to mine the rules. While a k value can be set to limit the number of rules generated, this does not appear to be a hard limit, especially for the fake news rules. Thus, to avoid creating an extremely large set of rules, different k values were used depending on the rules being mined, as shown in Appendix E. This particular package only outputs rules with antecedents in ascending numerical order, effectively losing the original representation with multiple instances of the same emotion in the text. To overcome this issue, a position marker of ascending value, i.e. 0, 2, 4, 6, 8. . . was prepended to each emotion label in the rules. The -1 indicates the end of an itemset, all of which only have one item; the -2 terminates each sequence. After the rules were generated, the position markers were removed. Duplicate rules were then removed. Rules without position markers were also tested. The mining process is shown in Figure 4.

Sequence, class:	<{happiness}{fear}{happiness}{disgust}{conj.}{happiness}>, real
Input to Miner:	002 -1 206 -1 402 -1 607 -1 37 -1 802 -1 1 -1 -2
Mined Rule:	206,802 ==>0 #SUP: 44 #CONF: 1.0
Final Rule:	6,2 ==>1

Figure 3.4: Class Sequential Rule Mining

Emotion sequences were generated from training and test posts with a method similar to that used to create the mining input rule sequences. The antecedent of a rule and the emotion sequence were compared using LCS, discussed under the k-medoids system. If the length of the LCS matched the length of the rule, the data sequence was considered to cover the rule, and the item corresponding to that rule was set in the rules vector. When generating training vectors, the posts were checked against rules from the matching category only. For the test vectors, each post was checked against every rule.

When all the vectors had been created, they were then concatenated. LIBSVM, available at `https://github.com/ocampor/libsvm` (Chang and Lin, 2011) was used to create a model from the training set. Tuning consisted of testing various settings, as discussed in Chapter 5. The design of this system did not require development data to be used in tuning. The final hyper-parameter configuration code is given in Appendix E. Finally, the test set was classified under that model. Further details on the SVM model and Top-k Class Sequential Rules algorithm are given in Chapter 4. Figure 5 shows the overall layout of this system.

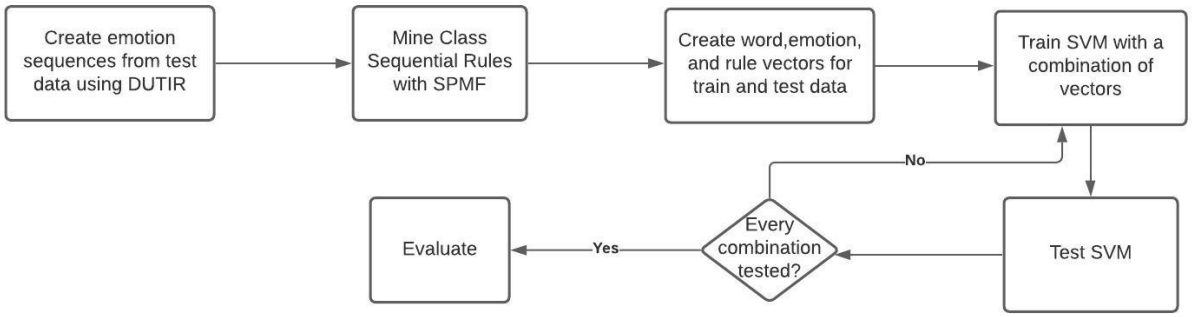

Figure 3.5: SVM Pipeline

3.3 Evaluation

The systems were evaluated on their classification ability using the metrics of precision, recall, and F1 score. Training in FlairNLP automatically included evaluation at the end, showing the values for these metrics. The other systems only provided the raw results, so the metrics were calculated using the confusion matrix provided in the scikit-learn package (`https://github.com/scikit-learn/scikit-learn`). The calculations are as follows:

$$Precision = \frac{correct\ \ system\ \ predictions}{system\ \ predictions} \tag{3.9}$$

Precision gives a picture of the quality of a system's predictions.

$$Recall = \frac{correct\ \ system\ \ predictions}{gold\ \ standard} \tag{3.10}$$

Recall shows how well a system's predictions stack up against the actual categories, the gold standard.

$$F_1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{3.11}$$

F1 score is a balance of sorts between precision and recall. These evaluation results are compared and discussed in Chapter 5. Additionally, listings of misclassified examples were used to create error vectors for further analysis in Chapter 6.

Chapter 4

ALGORITHMS

This chapter is intended to give a brief overview of the models and their associated algorithms so that the reader can understand their function within the context of this study. All models are well known within NLP, however, some specific adaptations have been made to suit the domain and task, as discussed in Chapter 3.

4.1 Neural Network

FlairNLP is built on Pytorch, an open-source machine learning library. The following section gives a technical overview of the Task-Aware Representation of Sentences (TARS) classifier and the different word embeddings used in this study.

Bidirectional Encoder Representations from Transformers (BERT) is a language representation model introduced by Devlin et al. (2019); the process is shown in Figure 1.

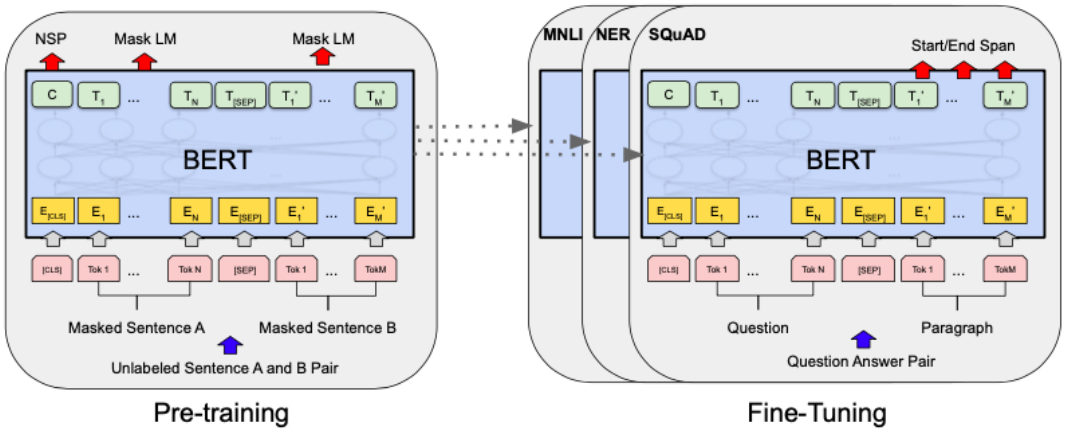

Figure 4.1: BERT Process (Devlin et al., 2019)

As the name suggests, BERT is strongly based on the Transformer architecture (Vaswani et al., 2017). The distinguishing feature of BERT is that it jointly conditions on both right and left context at every layer of the model. In other words, it is bidirectional, unlike some other models. In the BERT framework, there are two steps, namely, pre-training and fine-tuning. Pre-training requires unlabeled data, while fine-tuning uses labeled data from the task downstream from the model, which in this study is text classification.

Many of the pre-trained custom embeddings imported into FlairNLP are skip-gram. Introduced by Mikolov et al. (2013), skip-gram is a well-established word embedding. Skip-gram uses other words of a sentence to maximize the classification of the current word. The current word is input to a log-linear classifier having a continuous projection layer. The model works by predicting words with a set range both preceding and following the current word. More distant words are sampled less, giving them less weight because they typically have a weaker relationship with the current word than the words immediately surrounding it. Longer ranges produce better quality vectors, however, the tradeoff is longer training times.

The other custom embeddings developed specifically for Chinese are mostly based on Continuous Bag of Words (CBOW), also introduced by Mikolov et al. (2013). Similar to the standard bag of words approach, the order of the words is not particularly important since words are projected to the same position and the average of word vectors is then taken. However, unlike the standard bag of words approach, context uses a continuous distributed representation. Further details on these embeddings are given in Chapter 2.

FastText embeddings were introduced by Bojanowski et al. (2017). These word embeddings are based on the skip-gram model previously described. A bag of character n-grams represents a word. In turn, each character n-gram is tied to a vector representation; the final word representation combines all these representations as a sum. The strengths of FastText are accounting for subword information, handling rare/out of vocabulary words, and fast training time, as the name implies.

TARS is a native FlairNLP classifier particularly well-suited to working with one-shot

or zero-shot data. Developed by Halder et al. (2020), the architecture is transformer-based. Multiple tasks X,Y, and Z reflect the ability of the same model to be used across any task. A classification problem is then formulated as a query where the Task Label (classification) and Text pair are presented to the transformer. The transformer then predicts true/false regarding whether the label does indeed match the text. The representation of text and label are combined using a cross attention mechanism. TARS architecture is shown in Figure 2.

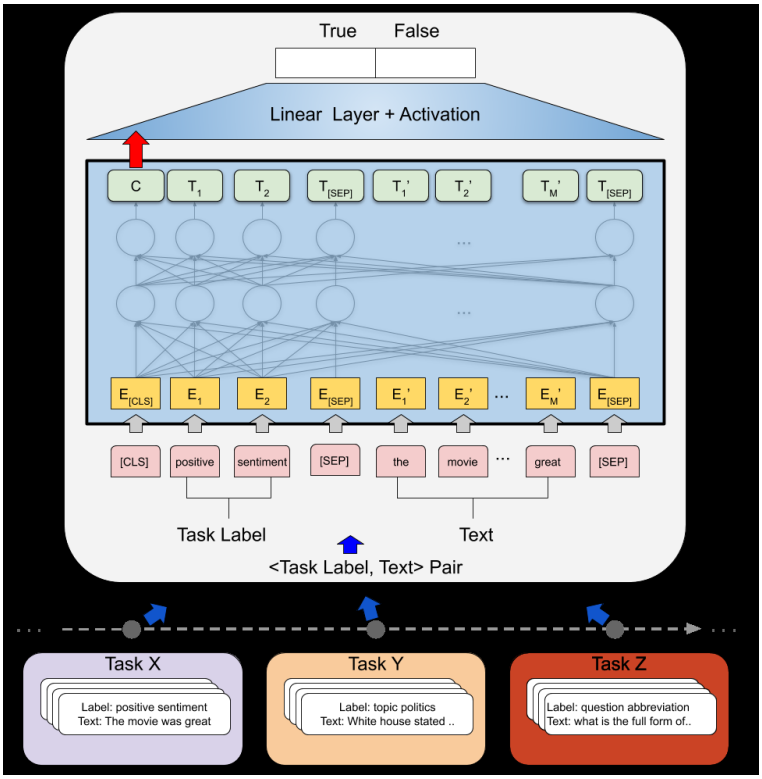

Figure 4.2: TARS Architecture (Halder et al., 2020)

4.2 HMM

HMMs consist of several key components: hidden states, transition probabilities, output probabilities, possible observations, and initial probabilities. Except the last one, these components are shown in Figure 3. The HMM used in this study has a somewhat different architecture from the HMMs typically used in NLP for sequencing tasks. Here the actual sequences are not important. Rather, the focus is on probabilities returned by the model.

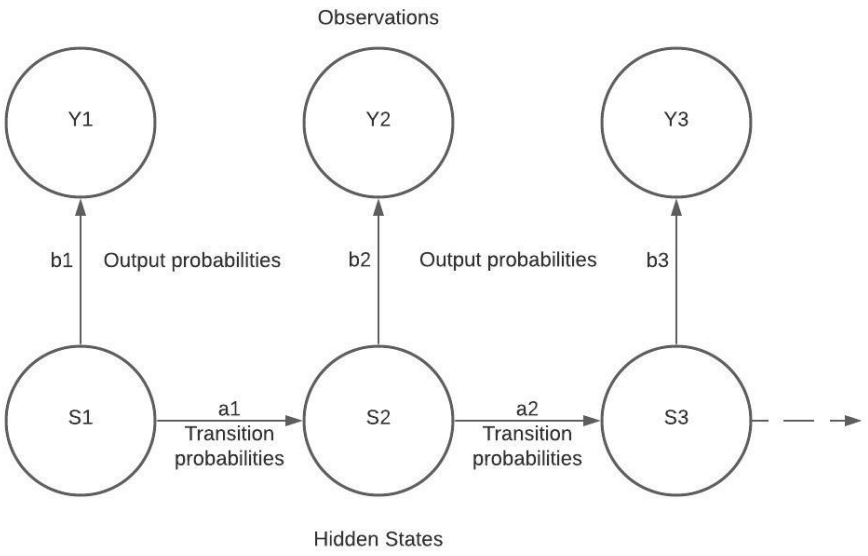

Figure 4.3: HMM Architecture

Observations are based on the five statistical features discussed in Chapter 3. These observations have a one-to-one mapping to the hidden states. Hidden states can be conceptualized as value sets representing the categories of fake or real news according to values of the statistical measures calculated from the training data. The output probabilities specify that only the current state s_p can give the observation y_k:

$$P(y_k|S_k = s_p) \tag{4.1}$$

Initial probabilities allow the state representing the first feature as the only possible starting state. Also, the transition probabilities specify that only transitions from the immediately preceding state s_q to the current state s_p are allowed:

$$P(S_k = s_p | S_{k-1} = s_q) = \begin{cases} 1 & (p = q + 1) \\ 0 & (p \neq q + 1) \end{cases} \tag{4.2}$$

Particle Swarm Optimization (PSO) was used for tuning feature weights for the models. PSO can be seen as a simulation of social behavior among living organisms, such as flocks of birds. Appendix B gives the pseudocode for the PSO algorithm, showing the process of updating position and velocity, while Figure 4 shows one iteration of PSO about halfway to convergence.

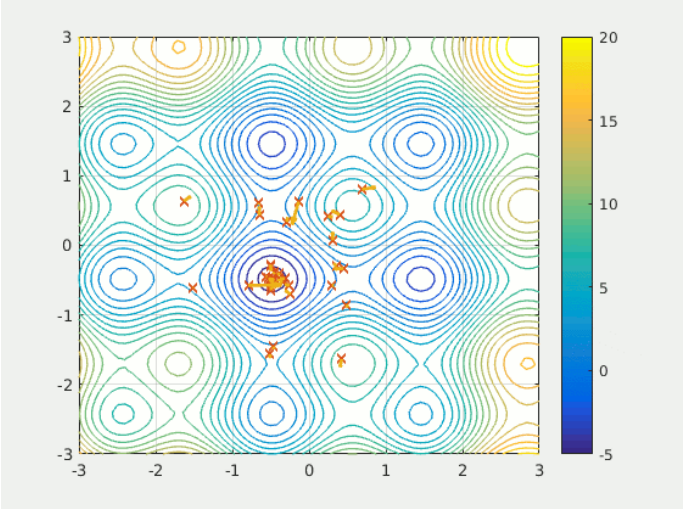

Figure 4.4: A PSO iteration "ParticleSwarmArrowsAnimation" by Ephramac is licensed under CC BY-SA 4.0 https://creativecommons.org/licenses/by-sa/4.0/

Here, some particles have converged on a solution, and the remaining particles are moving in closer. Through an iterative process, an optimal solution set can be found from the many candidate solutions, known as particles. However, there is no guarantee that the best solution will be found.

4.3 K-medoids

This algorithm is significantly different from the others in this study because it does not require training. Rather the goal is to classify data into n clusters, with n=2 in this study, representing the categories of real and fake news. The algorithm itself is quite straightforward and can be understood as a series of steps shown in Figure 5.

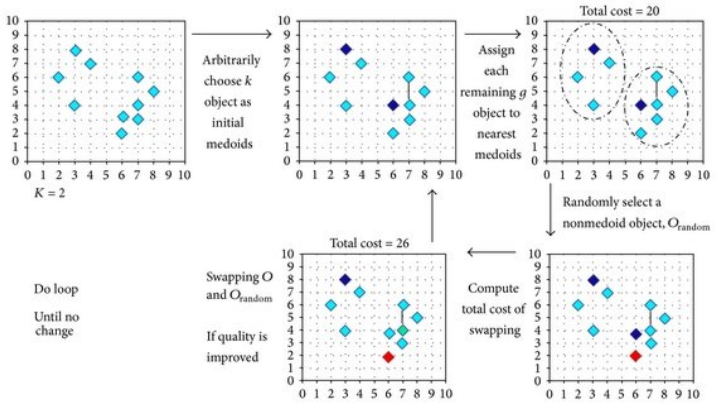

Figure 4.5: K-medoids "k-medoids clustering" is licensed under CC BY 3.0 https://creativecommons.org/licenses/by/3.0/

- Step 1: Select n medoids, typically chosen randomly. However, seeded values for initial medoids can also be used.
- Step 2: Assign each remaining example to the cluster where the distance from its medoid is the shortest.
- Step 3: Calculate the total distance of every member of each cluster from all other members of the same cluster. Then assign the member having the shortest total distance as the new medoid of that cluster.
- Step 4: Repeat steps 2 and 3 until the medoids do not update.

The distance used in steps 2 and 3 is either one of various distance measures for vectors (discussed in Chapter 5) or longest common subsequence (LCS). The algorithm for determining LCS is given in Appendix C, while an example is given below in Figure 6. The lengths

String 1:	1	0100	10	00	11		1		11	
String 2:	1	10	10	111	11	0	1	0	11	0
LCS:	1		10		11		1		11	

Figure 4.6: Longest Common Subsequence

and backtraces are stored so the substring can be recreated. However, in this system, only the substring length is needed. The substring itself is not significant because the focus is on how much sentiment the strings have in common, not the exact sentiment which the strings share.

4.4 SVM

The SVM model attempts to categorize data into a group based on the hyperplane boundary between them, as shown in Figure 6. Data are represented as points; those points appearing closest to the hyperplane serve as support vectors, hence the name of the model. SVM tries to establish which vectors represent examples with the most similarities, rather than ones with the most differences. The process of running the SVM is as follows:

- Step 1: Select two hyperplanes separating the data having no points between them.
- Step 2: Adjust the hyperplanes to maximize margin (orange area).
- Step 3: Set the average (red) line between the two hyperplanes as a decision boundary.

Here, the data on a 2D plane are linearly separable; in the case of non-linearly separable data, more dimensions are needed. Mapping the data to higher dimensions is done with the

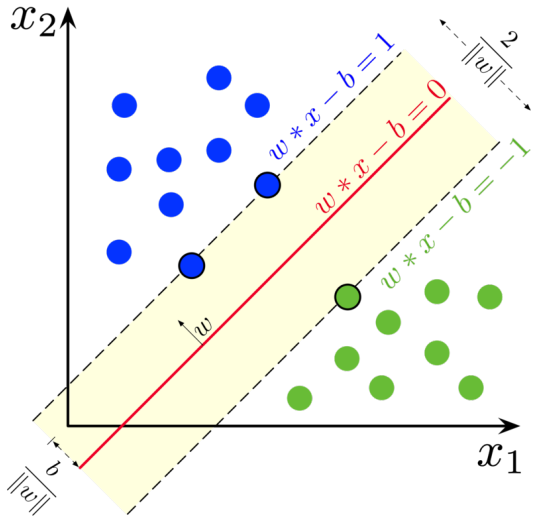

Figure 4.7: SVM "SVM Margin" by Larhmam is licensed under CC BY-SA 4.0 https://creativecommons.org/licenses/by-sa/4.0/

kernel method. Running a non-linear kernel SVM can be computationally intensive, leading to long run times on bigger datasets; fortunately, the dataset is this study in not very large.

In addition to word and emotion vectors, rule vectors are also used. Creation of these vectors first requires a rule miner to create class sequential rules from strings of sentiment and conjunctions found in the training data. To avoid creating many rules which may not have a good representation in the data, a top-k rules mining algorithm is used, shown in Appendix D.

Typically, both the antecedent and consequent can be expanded. However, for this study, the consequent has a single item–the category and only the antecedent contains multiple items. The key parameters for arriving at the final set of top k matching rules include the number of rules k, *confidence*, and *support*. *Confidence* indicates the number of sequences that satisfy the rule divided by the number of sequences matching the antecedent, but not necessarily the consequent. *Support* simply indicates the number of sequences satisfying the rule divided by the total sequences being mined.

Chapter 5

EXPERIMENTS

5.1 Neural Network

FlairNLP started giving fairly good results right out of the box. Some of the highest results are achieved by the pre-trained Chinese BERT and FastText embeddings, as well as the TARS classifier; these are all native to FlairNLP. While FlairNLP is relatively easy to use, a few challenges were encountered during the experiments. FastText web crawled embeddings are rather memory intensive, and caused the Google Colab environment to crash. This issue was resolved after upgrading to Colab PRO environment. Also, it should be noted that due to the nature of the neural network architecture, scores have the potential to vary somewhat between runs. This variance is more noticeable for fake news, due to the imbalanced ratio of real news to fake news.

Further testing on a segmented vs. unsegmented dataset showed the best results came from using the segmented one, which is the same data being passed to the other three models. F1 scores for real news are in a tight range of 0.97-0.98, while fake news F1 scores span a much wider range of 0.76-0.90. While a number of great results came from pre-trained skip-gram embeddings, training custom embeddings on a Weibo corpus also produced excellent results. The best results overall are from JWE (Joint Word Embeddings) word vectors.

As for fine tuning, the `reproject_word` parameter gave highest results when set to *true*. Adjusting the `hidden_size` and `reproject_words_dimension` document embedding parameters to various values such as 256, 512, and 1024 did not given any consistent improvements. Using embeddings in combination, also known as stacking, did not prove to be as helpful as initially anticipated. Sometimes stacking appeared to actually lower accuracy. Due to the large number of results, only results with an accuracy of 0.94-0.96 are shown in Table 1. The

results are grouped according to category, with P indicating precision, R indicating recall, and F1 indicating the F1 score, as discussed in Chapter 3.

	Real			Fake		
Embeddings/Classifier	P	R	F1	P	R	F1
TARS	0.97	0.98	0.98	0.89	0.80	0.84
BERT	0.98	0.99	0.98	0.92	0.85	0.88
FastText (Web Crawled)	0.94	1.00	0.97	0.96	0.63	0.76
Weibo GWE CBOW	0.96	0.99	0.98	0.94	0.75	0.83
Weibo JWE Word CBOW	0.98	0.99	**0.98**	0.92	0.88	**0.90**
Weibo JWE WordCharComponent CBOW	0.98	0.98	0.98	0.87	0.85	0.86
Weibo RECWE CBOW	0.97	1.00	0.98	0.97	0.78	0.86
Weibo RECWE char CBOW	0.97	0.99	0.98	0.91	0.78	0.84
Weibo skip-gram	0.95	0.99	0.97	0.93	0.70	0.80
Merged skip-gram	0.99	0.97	0.98	0.84	0.93	0.88
Chinese literature skip-gram	0.95	0.99	0.97	0.93	0.65	0.76
Chinese Wikipedia skip-gram	0.98	0.98	0.98	0.89	0.85	0.87
People's Daily News skip-gram	0.97	0.99	0.98	0.94	0.83	0.88
ZhiHu Question/Answer skip-gram	0.97	0.98	0.97	0.84	0.80	0.82
Baidu Baike Encyclopedia skip-gram	0.97	0.97	0.97	0.83	0.83	0.83

Table 5.1: Neural Network Results

5.2 HMM

Different types of HMMs and parameters were tested, with the Gaussian HMM and covariance type of *full* giving the best results overall. Both map and viterbi decoder algorithms were tested, with viterbi giving the most results having top accuracy. Every combination of different features was run to find the highest accuracy. Given the large number of feature combinations, only those combinations with an accuracy of 0.98 or greater are presented in Table 2.

		Real			Fake	
Features	P	R	F1	P	R	F1
TF-IDF+ECE	0.98	1.00	0.99	1.00	0.88	0.93
TF-IDF+ECE+IG	0.98	1.00	0.99	1.00	0.88	0.93
CHI+TF-IDF+ECE	0.99	1.00	0.99	0.97	0.93	0.95
CHI+TF-IDF+ECE+IG	0.99	1.00	**0.99**	1.00	0.93	**0.96**
MI+CHI	0.97	1.00	0.99	1.00	0.82	0.90
MI+TF-IDF	0.98	1.00	0.99	0.97	0.90	0.94
MI+TF-IDF+IG	0.98	1.00	0.99	1.00	0.90	0.95
MI+TF-IDF+ECE	0.98	1.00	0.99	1.00	0.90	0.95
MI+TF-IDF+ECE+IG	0.99	1.00	**0.99**	1.00	0.93	**0.96**
MI+CHI+IG	0.98	1.00	0.99	0.97	0.88	0.92
MI+CHI+ECE	0.99	0.98	0.99	0.88	0.95	0.92
MI+CHI+ECE+IG	0.98	1.00	0.99	1.00	0.90	0.95
MI+CHI+TF-IDF	0.99	.99	0.99	0.95	0.93	0.94
MI+CHI+TF-IDF+IG	0.99	1.00	**0.99**	1.00	0.93	**0.96**
MI+CHI+TF-IDF+ECE	0.99	1.00	0.99	0.97	0.93	0.95
MI+CHI+TF-IDF+ECE+IG	0.99	1.00	0.99	0.97	0.93	0.95

Table 5.2: HMM Results

Results show that TF-IDF and MI appear in more combinations than other features do. IG, the new feature added in this study, appears least in combination with other features. However, IG appears together with TF-IDF in all the combinations which have the highest F1 scores for both real and fake news. F1 values for real news are all 0.99, while F1 values for fake news range from 0.92 to 0.96, showing fake news is still challenging even for high-performing systems. During preliminary testing, the HMM system gave the most accurate results. However, after the final data were introduced and all the systems were tuned at a fine level, the HMM system and SVM system were tied for the highest accuracy.

Tuning the HMM system involved adding development data and adjusting feature weights using Particle Swarm Optimization (PSO), as discussed in Chapter 3. The tuning strategy was different for these two methods. PSO weights were used for every test run, while test runs before and after adding development data were compared to find the highest accuracy, resulting in a cycle of train-test (PSO)-tune (dev data)-test (PSO). Interestingly, tuning with development data sometimes, but not always gives better results than not tuning. As for PSO tuning, iterations of 10 and 100 were tried. Unsurprisingly, the 100 iterations test gave better results. By itself, the HMM does not require much time to build. However, 31 feature combination tests run through 100 iterations of PSO is 3100 iterations in total, giving this system the longest run time of all four. PSO Weights for top results are shown in Table 3.

Features	Weights
CHI+TF-IDF+ECE+IG	0.16656107435543055, 0.5378606575872462, 0.4749572600747287, 0.37012756704656535
MI+TF-IDF+ECE+IG	0.2894181439736472, 0.6891761769647031, 0.1711731330447952, 0.6091702620853043
MI+CHI+TF-IDF+IG	0.935159748353341, 0.1437767503275721, 0.3727296485801751, 0.007558850229457903

Table 5.3: PSO Weights for Top Results

5.3 K-Medoids

While the k-Medoids system was rather simple to create, tuning it provided more of a challenge. Under the original specifications of the model which allow for randomly chosen initial medoids, the results tended to fluctuate between runs. This was likely because the model converged to local optima which were not necessarily the global optima. The issue was resolved by setting the seed medoids to be the two examples which were the greatest distance apart according to the current distance measure being tested. One experiment compared 10 runs of randomly initialized medoids using cosine distance with the seeded medoids. The overall accuracy was similar when considering the highest scoring runs, and thus only the seeded medoids results are presented and discussed here.

Another part of tuning involved testing different distance measures for the word, emotion, and emotion intensity vectors to find the most optimal results. Chebyshev, Manhattan, and Euclidean distance measures all gave the highest overall accuracy scores on word, emotion, and emotion intensity vectors, performing very well in the real news category. Although the other distance measures usually gave better fake news F1 scores for word, emotion, and emotion intensity vector representations, the F1 scores for real news were lower, as was the overall accuracy. Also, constraining the distance values to be either 1 or 0 improved some distance measure results, and left others unchanged with the exception of correlation distance. A suitable alternative to LCS distances measure for the substrings was not found, so the emotion sequence results were not tuned further. The DUTIR sequences representation F1 score is fairly good for real news, although not the highest, while the ANTUSD sequences representation gets the highest F1 score for fake news overall.

Results of the experiments discussed above are shown in Table 4. They demonstrate that the k-medoids system is considerably better at real news classification than fake news classification. In fact, the F1 scores for fake news are zero whenever F1 scores for real news are the highest. The low ratio of fake news to real news allows for k-medoids to effectively fail to cluster the data but still achieve high F1 scores for a single category.

			Real			Fake	
Distance Measure	Representation	P	R	F1	P	R	F1
	Word	0.91	0.58	0.71	0.20	0.65	0.30
Bray-Curtis	Emotion	0.89	0.55	0.68	0.15	0.53	0.23
	Emotion Intensity	0.91	0.54	0.68	0.18	0.67	0.28
	Word	0.86	1.00	0.92	0.00	0.00	0.00
Canberra	Emotion	0.88	0.87	0.87	0.18	0.19	0.19
	Emotion Intensity	0.86	0.87	0.87	0.18	0.19	0.19
	Word	0.90	0.56	0.69	0.18	0.60	0.28
Correlation	Emotion	0.86	0.63	0.73	0.11	0.31	0.16
	Emotion Intensity	0.87	0.62	0.72	0.12	0.36	0.18
	Word	0.89	0.57	0.70	0.17	0.55	0.26
Cosine	Emotion	0.87	0.61	0.72	0.13	0.39	0.19
	Emotion Intensity	0.87	0.59	0.70	0.13	0.42	0.20
	Word	0.90	0.58	0.70	0.18	0.60	0.28
Jensen-Shannon	Emotion	0.87	0.54	0.67	0.13	0.47	0.21
	Emotion Intensity	0.87	0.52	0.65	0.13	0.47	0.20
Chebyshev,	word	0.86	1.00	0.92	0.00	0.00	0.00
Euclidean,	Emotion	0.87	0.99	**0.93**	0.00	0.00	0.00
Manhattan	Emotion Intensity	0.87	0.99	**0.93**	0.00	0.00	0.00
	HowNet Sequences	0.93	0.50	0.65	0.19	0.77	0.31
LCS Distance	ANTUSD Sequences	0.93	0.54	0.68	0.20	0.72	**0.32**
	DUTIR Sequences	0.91	0.57	0.70	0.18	0.64	0.28

Table 5.4: K-Medoids Results

5.4 SVM

During the tuning stage, multiple SVMs were tested, with C-SVM being found to be the most suitable for the type of data in this study. Of all the kernel types tried, the linear kernel gives the best results overall. Testing multiple C values from 1 to 4 did not give any change in results. Thus, the results presented in Table 5 are those from running a C-SVM with a linear kernel and C=1. All combinations of the emotion, rule, and word vectors were tested.

Tuning also focused on the class sequential rules, which are the most novel element of the system. The default setting for rules in this study is non-unique emotion sequences, meaning emotions can be repeated. Also, POS (conjunction, adverb, adjective) keywords are absent, and confidence=1.0. This confidence level indicates that sequences that satisfy a given rule are all from the same news category. Several variations of rules were tested to find the optimal format:

- Emotion+CONJ (conjunction), ADJ (adjective), and ADV (adverb) rules
- Emotion+CONJ rules with a lower confidence level of 0.75
- Emotion+CONJ rules with ascending placeholder values for uniqueness

Results show that while ADV rules outperform ADJ and CONJ rules when used alone, CONJ rules almost always work best with other vectors. While ascending placeholder values and a lower confidence level of 0.75 gave some limited improvements when the CONJ rule vector was used alone, the results did not improve when CONJ rule vectors are used in combination with other vectors, and thus those results have been omitted.

In the end, however, the highest F1 scores for both real and fake news are tied for the emotion+word combination and the emotion+word+rule (default) combination. Given that the same score can be achieved without rules, this suggests that rules are actually not very helpful in this system. Interestingly, emotion vectors do not work well alone but are helpful

in achieving the highest fake news F1 score when used in combination with word vectors. Overall, the highest F1 scores for fake and real news are tied for the HMM and SVM systems.

	Real			Fake		
Features	P	R	F1	P	R	F1
word	0.99	0.98	0.99	0.90	0.95	0.93
emotion	0.87	1.00	0.93	0.75	0.07	0.14
emotion+word	1.00	0.99	**0.99**	0.95	0.97	**0.96**
rule (default)	0.86	1.00	0.93	0.00	0.00	0.00
ADJrule	0.83	0.62	0.71	0.08	0.20	0.11
ADVrule	0.88	0.92	0.90	0.28	0.20	0.23
CONJrule	0.83	0.71	0.77	0.05	0.10	0.07
CONJrule (ascending)	0.81	0.41	0.55	0.10	0.40	0.16
CONJrule (confidence 0.75)	0.89	0.99	0.94	0.83	0.25	0.38
word+rule (default)	0.99	0.98	0.99	0.90	0.95	0.93
word+ADJrule	0.98	0.97	0.98	0.82	0.90	0.86
word+ADVrule	0.94	0.98	0.96	0.81	0.62	0.70
word+CONJrule	0.99	0.98	0.99	0.90	0.95	0.93
emotion+rule(default)	0.87	0.99	0.93	0.60	0.07	0.13
emotion+ADJrule	0.87	0.87	0.87	0.17	0.17	0.17
emotion+ADVrule	0.86	0.98	0.92	0.17	0.03	0.04
emotion+CONJrule	0.87	0.94	0.90	0.21	0.10	0.14
emotion+word+rule (default)	1.00	0.99	**0.99**	0.95	0.97	**0.96**
emotion+word+ADJrule	0.98	0.98	0.98	0.88	0.90	0.89
emotion+word+ADVrule	0.94	0.98	0.96	0.83	0.62	0.71
emotion+word+CONJrule	0.99	0.99	0.99	0.95	0.95	0.95

Table 5.5: SVM Results

5.5 Sentiment Findings

The original focus of this research was on the sentiment of fake news. However, in some systems, the sentiment behind fake news is not obvious. Rather, it is hidden in the layers of the neural network or abstracted away in the statistical measures of the HMM. The k-medoids and SVM systems, however, utilize the DUTIR sentiment lexicon, which does shed light on the actual sentiment of the news. The DUTIR lexicon gives seven main emotion categories along with intensity of the sentiment.

Averages were taken over the number of fake and real posts containing at least one emotion. They show happiness, like, fear, and disgust are stronger for real news, while anger, sadness, and surprise are stronger for fake news. Overall the highest and lowest averages are in the real news category, with the former belonging to like and fear, and the latter belonging to anger and surprise. The average value range for fake news emotions is narrower than that of real news. These averages are presented in Table 6.

Category	Happiness	Like	Anger	Sadness	Fear	Disgust	Surprise
real (emotion)	0.98	3.99	0.01	0.27	2.81	0.95	0.00
fake (emotion)	0.58	1.89	0.11	0.44	0.86	0.64	0.08
real (intensity)	4.88	17.80	0.09	1.32	13.67	4.0	0.02
fake (intensity)	3.14	9.11	0.78	2.0	4.08	2.58	0.58

Table 5.6: Average DUTIR Sentiment

The other two lexicons, HowNet and ANTUSD, are used only in the k-medoids system. Unlike DUTIR, only positive and negative polarity is captured, with fake news carrying higher average values than real news. This result agrees with the finding of Cui et al. (2019), discussed in Chapter 2. Generally, positive polarity is stronger than negative, with the interesting exception of ANTUSD for real news, where negative polarity is actually slightly stronger. The results are presented in Table 7.

Lexicon	Category	Positive	Negative
HowNet	Real	0.016	0.008
	Fake	0.128	0.026
ANTUSD	Real	0.008	0.012
	Fake	0.15	0.125

Table 5.7: Average HowNet and ANTUSD Sentiment

5.6 Overall Results

The F1 results for all the systems are shown together for comparison in Figure 1. Here, it can be clearly seen that all systems excel at classifying real news, with all the F1 scores above 0.9. Fake news is clearly more challenging. While HMM and SVM are tied, and Neural Network system is only slightly lower, the clear outlier is the k-medoids system. Although k-medoids failed to classify any fake news correctly, it did provide some insights into the actual sentiment of fake news. The error analysis addresses these outcomes in Chapter 6.

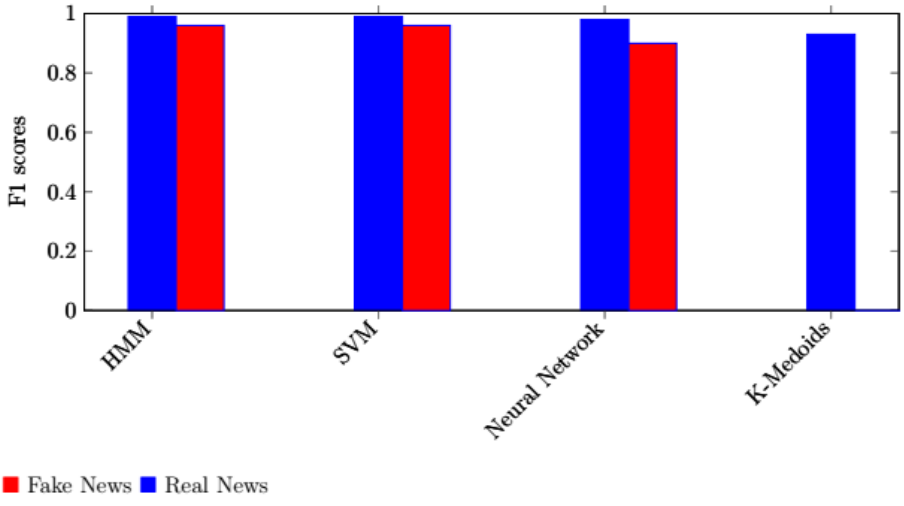

Figure 5.1: System F1 Scores Comparison by Category

Chapter 6

DISCUSSION

After the systems were fine tuned on the final dataset, the error analysis commenced. Each system produced an error vector of incorrectly classified Weibo posts. The error vectors from the best performing run of each system are presented in Table 1. During the first iteration of the error analysis, an incorrectly extracted post was discovered. This particular post contained an English translation with the word *who*, which matched the keyword WHO. The post was removed from the dataset, the data was re-partitioned and tests were rerun, showing a significant improvement in results. The following sections of this chapter explore the details of errors for each system, particularly in relation to other systems. For readability, the posts featured in this analysis are those from the raw extracted data, prior to preprocessing. The original Chinese post is given first, followed by an English translation. Translations were created initially by Google Translate, post-edited by the author, and finally checked for accuracy by a native speaker of Chinese who is also competent in English.

System	Fake News Errors	Real News Errors
K-medoids	0, 2, **7**, 17, **18**, 22, 23, 24, 26, 29, **48**, 53, 70, 72, 77, 85, 110, 118, 126, 151, 163, 173, 180, 181, **188**, 198, 221, **222**, **228**, 240, **245**, 255, **257**, 266, 284, 289	1,290
Neural Network	**7**, **18**, 48, 188, **222**, **228**, **257**	32, 33
HMM	**222**, **228**/245, **257**	
SVM	**257**	94, 211

Table 6.1: Errors by System and Category

6.1 Neural Network Error Analysis

Most errors are concentrated in the fake news category. The most interesting classification error overall is #257, which is also misclassified by all systems. Given the original post was rather long, a truncated version is given here:

#257 头三张图来自网友投稿，河南郑州地铁内公益广告在致敬抗疫英雄，全部为男性照片，无一女性面孔@郑州地铁抗疫期间，全国出征的女性医护占比90%。。。。

The first three pictures were contributed by netizens. The public service advertisements in Zhengzhou Metro, Henan Province pay tribute to the anti-epidemic heroes. They are all male photos, and there is no female face. @Zhengzhou Metro During the anti-epidemic period, 90% medical nurses recruited nationwide were female. . . .

Further analysis found that the COVID-19 keyword is 抗疫, *anti-epidemic*, which appears over 350 times in the training dataset. It corresponds to real news around 96% of the time, which is higher than the actual percentage of real news in the training dataset. Another key expression, 郑州地铁, *Zhengzhou Metro* does not appear in training data. Finally, this example has photos, which are not considered by this system.

Other errors in common with the HMM system and k-medoids system are discussed under the error analysis of those systems. As for the neural network specific errors, the posts are given in Appendix F and discussed here. #32 and #33 both are both misclassifications of real news. Both of these posts contain keywords common to both categories, and the first is rather unremarkable. The second, however, is a reference to satire, which is quite challenging for this system to detect. Errors #48 and #118 are misclassifications of fake news. In the first post, the keyword 试剂盒 *testing kit* seldom appears in training data, which might present a difficulty to this system. 张文宏 *Zhang Wenhong*, from the second post is somewhat more common in the training data, showing names can present a challenge to this system, perhaps due to their propensity to appear in both real and fake news.

6.2 HMM Error Analysis

Unlike the other systems, all HMM errors are fake news misclassifications. The HMM system had two runs with different results tie for best performance. The errors from both runs are examined here. Errors #222 and #257 are common to both runs and #228 is from one run of the system. Error #257 was discussed in the previous section. Errors #222 and #228, in common with the neural network system and k-medoids systems, are as follows:

#222 据说国际航班全部交给上海执行，上海成为唯一国际口岸#中医##健康达人记##中医抗疫第一线

It is said that all international flights are handed over to Shanghai, and Shanghai has become the only international port.#Chinese Medicine##Health Expert## TCM, the first line of combating the disease

This post has rather innocuous wording. 国际, *international* appears more in real news (97%) than fake news, while 口岸, *port* follows the distribution of the categories, and is thus not helpful. The second post was truncated due to being quite lengthy.

#228——白宫官网消息：吉利德公司的瑞德西韦明日（2月3日）开始在中国临床试验。经特朗普总统特批，同意将该药物专利豁免，向中国紧急公开药物分子结构。允许中国仿制此药用于治疗冠状病毒。。。。

White House official website news: Gilead's Remdesivir will start clinical trials in China tomorrow (February 3). With the special approval of President Trump, it was agreed to grant a waiver of the drug patent and disclose the molecular structure of the drug to China in an expedited manner. This will allow China to imitate this medicine for the treatment of coronavirus. . . .

Interestingly, this second post does contain a useful keyword, 瑞德西韦, *Remdesivir*, which appears more frequently in fake news training data but was insufficient to make the system classify the post as fake. Another keyword,特朗普 *Trump*, shows a distribution more in line with the proportion of fake news to real news, and is not helpful in the classification. The final error, from one run of the system and in common with k-medoids is #245:

#245 北京疫情部分高速进京道路封闭注意绕行

Some highways entering Beijing due to epidemic are closed, pay attention to detours

Interestingly, both 北京疫情, *Beijing Epidemic* and 封闭, *closed* appear more frequently in fake news, while 绕行, *detour* does not appear in the training data at all.

6.3 K-medoids Error Analysis

Although the majority of errors for k-medoids, neural network, and HMM are misclassified fake news, k-medoids really does not have much in common with the other systems. The run with the highest accuracy completely failed to classify any fake news correctly. While keywords for other systems are analyzed based on their counts in the training data, for k-medoids, analysis is based on test data because training data is not used.

The errors in common with the neural network system are #7 and #18, given as follows:

#7 学校开学时间：高三，初三，3月2日开学，高一，高二，初一，初二，3月9日开学，小学4—6年级，3月16日开学，幼儿园，小学1—3年级，大学，职校继续延时，直至疫情消除。市政府会议精神，3月16日居民出行正常化，17日公交正常化，18日逐步企业生产和市场经营正常化，22日重点场所正常化，25日机场、高速、动车、国道正常化。目前为止待在家里！等待通知！以上是真的嘛??第一次无比怀念在学校的日子，我留下了不学无术的泪水。

School start time: Grade 12 and Grade 9, start on March 2; Grade 10, Grade 11, Grade 7, and Grade 8 start on March 9; elementary school grades 4 to 6 start on March 16; kindergarten, elementary school grades 1 to 3, universities, and vocational schools continue to delay reopening until the epidemic is eliminated. In the spirit of the city government meeting, residents' travel was normalized on March 16 and public transportation on the 17th. Enterprise production and market operations were gradually normalized on the 18th, key locations on the 22nd, and airports, highways, high-speed trains, and national highways on the 25th. Staying at home so far! Waiting for notification! Is the above true?? This is the first time I missed school days so much. I wept tears over my lack of learning.

Ironically, there are three instances of the expression 学校开学时间 *school start time* in the test data, but only this example is classified incorrectly. The keyword 疫情, *pandemic* is not useful because it could be expected that both categories would have this keyword in common. In fact, the percentages of this keyword in each category are quite similar to the percentages of categories themselves as part of the dataset.

#18 #北京疫情扩散风险很高#，北京疫情严峻，84岁的钟南山院士再次出征北京，他就是最耀眼的一颗星，为钟老点赞！#北京要求最短时间内找到所有密接者#

#The risk of the spread of the epidemic in Beijing is very high#, and the epidemic situation in Beijing is grim. The 84-year-old Academician Zhong Nanshan once again set off for Beijing. He is the most dazzling star, let's all praise Mr. Zhong! #Beijing requires all the close contacts (of infected persons) to be found in the shortest time

Errors #26 and #85 (fake) and error #290 (real) also reference 钟南山 *Zhong Nanshan*:

#290【人民微评：敢医敢言钟南山】#钟南山呼吁公众继续保持距离#，言辞恳切，娓娓道来，深入浅出，入耳入心。这不是钟南山院士第一次提醒，别当耳旁风，最好的尊重是听从。“请离我远点”，是一种善意；“我离你远点”，是一种责任。非常时期，就该适当保持距离。距离产生美，也产生安全感。

[People's microblog-comment: Zhong Nanshan dares to treat, dares to speak] #Zhong Nanshan appeals to the public to keep their distance#, speaking sincerely and clearly explaining the situation in simple terms. Take these matters to heart. This is not the first time that Academician Zhong Nanshan has reminded us. Don't be deaf to his words, the best form of respect is to listen to him. "Please stay away from me", is a gesture of goodwill; "I stay away from you" is a responsibility. In extraordinary times, you should keep a proper distance. Distance produces beauty and also a sense of security.

The name 钟南山, *Zhong Nanshan* appears proportionately in both real news and fake news, and as shown in these examples, is not helpful in making the correct classification. Given that no fake news examples were classified correctly, the remaining fake news misclassifications unique to k-medoids are not discussed. As for real news, the only other example, #1 is actually the medoid of the fake news group:

#1【谢谢你们，“浆”爱传递】“#我是被救者我想去救人#”“虽然不能上一线，但希望我的血浆能拯救更多患者”...谢谢你们，恢复期血浆捐献者。挺身而出的你们，让爱与希望，层层传递。这场战“疫”，团结的我们一定能赢，#谢谢每个平凡的中国人#！

[Thank you, pass on the "plasma" love] "#I was saved (by plasma treatments); now, I want to save others#" "Although I can't be on the front line, I hope my plasma can save more patients"...Thank you, convalescent plasma donors. You who are coming forward, let love and hope be passed on down to others. In this "epidemic" war, we who are united will surely win, #thanks to each ordinary Chinese person#!

The distance between this post and other posts might arise from 血浆, *blood plasma*, which in the test data appears only in this one example. Overall, these results suggest that the unsupervised approach of k-medoids is unsuitable for this particular classification task, at least without significant modifications.

6.4 SVM Error Analysis

Error #257 is shared with all systems as previously discussed. Errors #94 and #211, unique to the SVM system, are both examples of misclassified real news. The majority of SVM errors involve real news; this error distribution could be a result of differences in how the classifier works relative to the other systems.

#94 【#文旅部提醒切勿前往澳大利亚旅游#：种族歧视和暴力上升】6月5日，文化和旅游部发布赴澳大利亚旅游安全提醒，近期，由于受到新冠肺炎疫情影响，澳大利亚国内对华人和亚裔的种族歧视言行和暴力行为现象明显上升。文化和旅游部提醒中国游客切实提高安全防范意识，切勿前往澳大利亚旅游。O网页链接

The Ministry of Culture and Tourism reminds you not to travel to Australia due to the rise of racial discrimination and violence. The Ministry of Culture and Tourism issued a safety reminder for travel to Australia. Recently, due to the COVID-19 epidemic, the phenomenon of racial discrimination and violence against Chinese and Asians by Australians has increased significantly. The Ministry of Culture and Tourism reminds Chinese tourists to effectively raise their awareness of safety precautions and not to travel to Australia. O web link

文旅部, *The Ministry of Culture and Tourism*, does not appear in the training data. Interestingly, 澳大利亚, *Australia* only appears in real news training data. The remaining keywords, referring to COVID-19 itself, are not good discriminators, as previously mentioned.

#211【#钟南山强调重视入境人员检测#】钟南山院士说，我们国内的病情基本稳定住了，但是其他国家的发展不容乐观；而且病毒的潜伏期因人而异，我们必须要对入境的外籍人员进行必要的检测。LCGTN记者团的微博视频

Zhong Nanshan emphasizes the importance of immigration inspection. Academician Zhong Nanshan said that our domestic condition is basically stable, but new developments in other countries are not optimistic. Also, the incubation period of the virus varies from person to person. We must carry out necessary tests on foreigners entering the country. Weibo video of LCGTN reporter team

Interestingly, keywords 潜伏期, *incubation period* and 外籍人员, *foreigners* appear only in real training data, but were not helpful here. Also, the difficulty of classifying posts with the name 钟南山, *Zhong Nanshan* is again apparent, as discussed previously.

Chapter 7

CONCLUSION

Going into this study, one initial concern was dealing with the character encodings for Chinese. However, encoding turned out to not be a major issue, with most resources and data using UTF-8 encoding. Whenever encoding issues did appear, they were usually easy to resolve, with only one case of encoding issues preventing the utilization of word embeddings.

Unsurprisingly, one major challenge turned out to be learning to use code packages with varying degrees of documentation. The FlairNLP system was the easiest to implement overall, thanks to good documentation. It was initially anticipated that this system would dominate, considering its record of achieving new best accuracy scores on various NLP tasks. However, experiments showed that not only did SVM and HMM both outperform FlairNLP, but in an interesting twist, their top results were tied. Although k-medoids was the lowest ranking system, it provided the most insight of any system on the test data sentiment.

Once the systems were initially implemented, tuning them was another challenge. The iterative tuning and testing process most likely took the majority of the development time. Error analysis provided a chance to gain some final insights into the strengths and weaknesses of the systems. The challenge here was trying to determine what the machine might be using to make its classification, not just using human perception.

One limitation of the dataset is that all real posts come from an official news source; there are no real posts from ordinary people. In future studies, it would be useful to analyze real posts from different sources to determine the influence of different writing styles on real news classification. Also, some COVID-19 keywords were found to be segmented incorrectly; lexicons could be updated to ensure proper segmentation for future classification systems.

BIBLIOGRAPHY

Ahmad, I., Yousaf, M., Yousaf, S., and Ahmad, M. O. (2020). Fake news detection using machine learning ensemble methods. *Complexity*, 2020.

Ajao, O., Bhowmik, D., and Zargari, S. (2019). Sentiment aware fake news detection on online social networks. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2507–2511. IEEE.

Akbik, A., Bergmann, T., Blythe, D., Rasul, K., Schweter, S., and Vollgraf, R. (2019). Flair: An easy-to-use framework for state-of-the-art nlp. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59.

Bhutani, B., Rastogi, N., Sehgal, P., and Purwar, A. (2019). Fake news detection using sentiment analysis. In *2019 Twelfth International Conference on Contemporary Computing (IC3)*, pages 1–5. IEEE Computer Society.

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

Cambria, E. and Hussain, A. (2015). *Sentic computing: A Common-Sense-Based Framework for Concept-Level Sentiment Analysis*. Springer International Publishing.

Cao, S., Lu, W., Zhou, J., and Li, X. (2018). Cw2vec: Learning chinese word embeddings with stroke n-gram information. In *Thirty-second AAAI Conference on Artificial Intelligence*.

Chang, C. C. and Lin, C. J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):1–27.

Chen, X., Xu, L., Liu, Z., Sun, M., and Luan, H. (2015a). Joint learning of character and word embeddings. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Chen, Z. and Hu, K. (2018). Radical enhanced chinese word embedding. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, pages 3–11. Springer.

Chen, Z., Huang, Y., Tian, J., Liu, X., Fu, K., and Huang, T. (2015b). Joint model for subsentence-level sentiment analysis with markov logic. *Journal of the Association for Information Science and Technology*, 66(9):1913–1922.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to Algorithms*. Cambridge, Mass, MIT Press.

Cui, L., Wang, S., and Lee, D. (2019). Same: sentiment-aware multi-modal embedding for detecting fake news. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 41–48.

Devlin, J., Chang, M. W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Dong, Z., Dong, Q., and Hao, C. (2010). Hownet and its computation of meaning. In *Coling 2010: Demonstrations*, pages 53–56.

Fournier-Viger, P. and Tseng, V. S. (2011). Mining top-k sequential rules. In *International Conference on Advanced Data Mining and Applications*, pages 180–194. Springer.

Halder, K., Akbik, A., Krapac, J., and Vollgraf, R. (2020). Task-aware representation of

sentences for generic text classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3202–3213.

He, Y., Alani, H., and Zhou, D. (2010). Exploring english lexicon knowledge for chinese sentiment analysis. In *CIPS-SIGHAN Joint Conference on Chinese Language Processing*.

Hudaib, A. A. and Hwaitat, A. (2017). Movement particle swarm optimization algorithm. *Modern Applied Science*, 12(1):148.

Kula, S., Choraś, M., Kozik, R., Ksieniewicz, P., and Woźniak, M. (2020). Sentiment analysis for fake news detection by means of neural networks. In *International Conference on Computational Science*, pages 653–666. Springer.

Li, C., Xu, B., Wu, G., He, S., Tian, G., and Hao, H. (2014). Recursive deep learning for sentiment analysis over social data. In *2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, volume 2, pages 180–185. IEEE.

Li, S., Zhao, Z., Hu, R., Li, W., Liu, T., and Du, X. (2018). Analogical reasoning on chinese morphological and semantic relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 138–143.

Liu, L., Lei, M., and Wang, H. (2013). Combining domain-specific sentiment lexicon with hownet for chinese sentiment analysis. *J. Comput.*, 8(4):878–883.

Liu, L., Luo, D., Liu, M., Zhong, J., Wei, Y., and Sun, L. (2015). A self-adaptive hidden markov model for emotion classification in chinese microblogs. *Mathematical Problems in Engineering*, 2015.

Liu, P., Zhang, J., Leung, C. W. K., He, C., and Griffiths, T. L. (2018). Exploiting effective representations for chinese sentiment analysis using a multi-channel convolutional neural network. *arXiv preprint arXiv:1808.02961*.

Liu, S. M. and Chen, J. H. (2015). A multi-label classification based approach for sentiment classification. *Expert Systems with Applications*, 42(3):1083–1093.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Peng, H. and Cambria, E. (2017). Csenticnet: a concept-level resource for sentiment analysis in chinese language. In *International Conference on Computational Linguistics and Intelligent Text Processing*, pages 90–104. Springer.

Peng, H., Cambria, E., and Hussain, A. (2017). A review of sentiment analysis research in chinese language. *Cognitive Computation*, 9(4):423–435.

Quan, C., Wei, X., and Ren, F. (2013). Combine sentiment lexicon and dependency parsing for sentiment classification. In *Proceedings of the 2013 IEEE/SICE International Symposium on System Integration*, pages 100–104. IEEE.

Su, T.-R. and Lee, H.-Y. (2017). Learning chinese word representations from glyphs of characters. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 264–273.

Su, Z., Xu, H., Zhang, D., and Xu, Y. (2014). Chinese sentiment classification using a neural network tool—word2vec. In *2014 International Conference on Multisensor Fusion and Information Integration for Intelligent Systems (MFI)*, pages 1–6. IEEE.

Tan, L. and Bond, F. (2011). Building and annotating the linguistically diverse ntu-mc (ntu-multilingual corpus). In *Proceedings of the 25th Pacific Asia Conference on Language, Information and Computation*, pages 362–371.

Tan, S. and Zhang, J. (2008). An empirical study of sentiment analysis for chinese documents. *Expert Systems With Applications*, 34(4):2622–2629.

Wan, X. (2008). Using bilingual knowledge and ensemble techniques for unsupervised chinese sentiment analysis. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 553–561.

Wei, B. and Pal, C. (2010). Cross lingual adaptation: an experiment on sentiment classifications. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 258–262.

Wei, G., An, H., Dong, T., and Li, H. (2014). A novel micro-blog sentiment analysis approach by longest common sequence and k-medoids. In *Proceeding of the 19th Pacific Asia Conference on Information Systems (PACIS 2014)*, page 38.

Wen, S. and Wan, X. (2014). Emotion classification in microblog texts using class sequential rules. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*.

Wu, S. J. and Chiang, R. D. (2015). Using syntactic rules to combine opinion elements in chinese opinion mining systems. *Journal of Convergence Information Technology*, 10(2):137.

Xiong, W., Jin, Y., and Liu, Z. (2014). Chinese sentiment analysis using appraiser-degree-negation combinations and pso. *J. Comput.*, 9(6):1410–1417.

Xu, H., Zhao, K., Qiu, L., and Hu, C. (2010). Expanding chinese sentiment dictionaries from large scale unlabeled corpus. In *Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation*, pages 301–310.

Xu, W., Liu, Z., Wang, T., and Liu, S. (2013). Sentiment recognition of online chinese micro movie reviews using multiple probabilistic reasoning model. *J. Comput.*, 8(8):1906–1911.

Yang, C., Zhou, X., and Zafarani, R. (2021). Checked: Chinese covid-19 fake news dataset. *Social Network Analysis and Mining*, 11(1):1–8.

Yu, J., Jian, X., Xin, H., and Song, Y. (2017). Joint embeddings of chinese words, characters, and fine-grained subcharacter components. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 286–291.

Yuan, B., Liu, Y., and Li, H. (2013). Sentiment classification in chinese microblogs: lexicon-based and learning-based approaches. *International Proceedings of Economics Development and Research*, 68:1.

Zagibalov, T. and Carroll, J. A. (2008). Unsupervised classification of sentiment and objectivity in chinese text. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume I*.

Zhai, Z., Xu, H., Kang, B., and Jia, P. (2011). Exploiting effective features for chinese sentiment classification. *Expert Systems with Applications*, 38(8):9139–9146.

Zhang, C., Zeng, D., Li, J., Wang, F.-Y., and Zuo, W. (2009). Sentiment analysis of chinese documents: From sentence to document level. *Journal of the American Society for Information Science and Technology*, 60(12):2474–2487.

Zhang, P. and He, Z. (2013). A weakly supervised approach to chinese sentiment classification using partitioned self-training. *Journal of Information Science*, 39(6):815–831.

Appendix A

DATASET KEYWORDS

Categories	Keyword (Chinese)	Translation (English)
Coronavirus and COVID-19	冠状病毒	Coronavirus
	新冠肺炎	COVID-19
	新冠	(Abbr.) Coronavirus/COVID-19
	Coronavirus	
	SARS-CoV-2	
	COVID	
Pandemic	疫情	Pandemic/epidemic
	疫区	Pandemic/epidemic area
	传染, 感染	Infection
	确诊	Confirmed case
	死亡病例	Death case
	输入病例, 输入性传播	Imported case
Figures and organizations	世界卫生组织	WHO
	世卫	(Abbr.) WHO
	钟南山	Nanshan Zhong
	张文宏	Wenhong Zhang
	李文亮	Wenliang Li
	福奇	Fauci
	WHO	
	CDC	
	试剂盒	Testing kit

	核酸检测	Nucleic Acid Test
	疫苗	Vaccine
Medical	抗体	Antibody
supplies	火神山	Huoshenshan
	雷神山	Leishenshan
	口罩	Mask
	N95	
	隔离	Quarantine
	封城	Lockdown
	防控	Prevention And Control
	群体免疫	Herd (community) immunity
	健康码, 健康宝	Health code
	战疫, 抗疫	Combat COVID-19
	援鄂	Love for Wuhan

Yang et al. (2021)

Appendix B

PARTICLE SWARM OPTIMIZATION PSEUDOCODE

```
For each particle
   Initialize particle
END
Do
   For each particle
      Calculate fitness value
      If the fitness value is better than the best fitness value (pBest) in history
         set current value as the new pBest
   End

   Choose the particle with the best fitness value of all the particles as the gBest
   For each particle
      Calculate particle velocity according equation          (2)
      Update particle position according equation             (1)
   End
While maximum iterations or minimum error criteria is not attained
```

Hudaib and Hwaitat (2017)

Appendix C

LONGEST COMMON SUBSEQUENCE PSEUDOCODE

```
LCS-LENGTH(X, Y)
 1  m = X.length
 2  n = Y.length
 3  let b[1..m, 1..n] and c[0..m, 0..n] be new tables
 4  for i = 1 to m
 5      c[i, 0] = 0
 6  for j = 0 to n
 7      c[0, j] = 0
 8  for i = 1 to m
 9      for j = 1 to n
10          if x_i == y_j
11              c[i, j] = c[i − 1, j − 1] + 1
12              b[i, j] = "↖"
13          elseif c[i − 1, j] ≥ c[i, j − 1]
14              c[i, j] = c[i − 1, j]
15              b[i, j] = "↑"
16          else c[i, j] = c[i, j − 1]
17              b[i, j] = "←"
18  return c and b
```

Cormen et al. (2009)

Appendix D

TOP-K SEQUENTIAL RULES PSEUDOCODE

```
TOPSEQRULES (S, k, minconf)
1.  R := Ø. L := Ø. minsup := 0.
2.  Scan the database S once. Record the sids set of each item c in a variable sids(c).
3.  FOR each pairs of items i, j such that |sids(i)| ≥ minsup and ∩ |sids(j)| ≥ minsup:
4.      sids(i⇒j) := Ø. sids(j⇒i) := Ø.
5.      FOR each sid s ∈ (sids(i) ∩ sids(j):
6.          IF i occurs before j in s THEN sids(i⇒j) := sids(i⇒j) ∪ {s}.
7.          IF j occurs before i in s THEN sids(j⇒i) := sids(j⇒i) ∪ {s}.
8.      END FOR
9.      IF |sids(i⇒j)| / |S| ≥ minsup THEN
10.         conf({i}⇒{j}) := |sids(i⇒j)| / | sids(i).
11.         IF conf({i}⇒{j}) ≥ minconf THEN SAVE({i}⇒{j}, L, k, minsup).
12.         Set flag expandLR of {i}⇒{j}to true.
13.         R := R∪{{i}⇒{j}}.
14.     END IF
…       [lines 9 to 14 are repeated here with i and j swapped]  …
15. END FOR
16. WHILE ∃r ∈ R AND sup(r) ≥ minsup DO
17.     Select the rule rule having the highest support in R
18.     IF rule.expandLR = true THEN
19.         EXPAND-L(rule, L, R, k, minsup, minconf).
20.         EXPAND-R(rule, L, R, k, minsup, minconf).
21.     ELSE EXPAND-R(rule, L, R, k, minsup, minconf).
22.     REMOVE rule from R. REMOVE from R all rules r ∈ R | sup(r) <minsup.
23. END WHILE
```

Fournier-Viger and Tseng (2011)

Appendix E

SYSTEMS CONFIGURATION AND HYPER-PARAMETERS

E.1 Neural Network

```
#TARS approach (using flair package)
tars = TARSClassifier(task_name='tarsChinese',
label_dictionary=corpus.make_label_dictionary())
trainer = ModelTrainer(tars, corpus)
trainer.train(base_path='/path/to/model',learning_rate=0.02,
mini_batch_size=16, mini_batch_chunk_size=4, max_epochs=25)
#Word Embeddings approach (using flair package)
word_embeddings = [bert_embedding, . . .]#list embeddings here
document_embeddings = DocumentRNNEmbeddings(word_embeddings, hidden_size=512,
reproject_words=True, reproject_words_dimension=512)
classifier = TextClassifier(document_embeddings,
label_dictionary=corpus.make_label_dictionary(), multi_label=False)
trainer = ModelTrainer(classifier, corpus)
trainer.train('./', max_epochs=25)
```

E.2 HMM

```
#create model with initial parameters (using hmmlearn package)
s = number_of_features_in_model #i.e. s=3; features MI, TF-IDF, CHI-Square
model = hmm.GaussianHMM(n_components=s, covariance_type="full", verbose=False)
model.startprob_ = np.zeros(s)
model.startprob_[0] = 1.0 #HMM can only start at the first feature
```

```
model.transmat_ = np.identity(s) #only allow transitions to the next state
model.fit(data)
#tuning hyper-parameters (using pyswarms package)
bounds = (np.zeros(s), np.ones(s)) #min and max allowed parameters
options = {'c1': 1, 'c2': 1, 'w':1} #c1 is cognitive, c2 is social, w is inertia
optimizer = GlobalBestPSO(n_particles=1, dimensions=s,
options=options, bounds=bounds)
cost, pos = optimizer.optimize(trainHMM, 100, featureSelection=fVec) #iterations
```

E.3 K-medoids

```
#tuning Manhattan distance measure; constrain the values to be between 0 and 1
distance = np.clip(spatial.distance.cityblock(vectors[x], vectors[y]), 0, 1)
```

E.4 SVM

```
#rule mining
#arguments: mine the top 15 rules with confidence of 100% (1) for fake news (0)
spmf = Spmf('TopSeqClassRules', input_filename='rules.txt',
output_filename='/path/to/file/fakeRules.txt',
spmf_bin_location_dir='/path/to/file/',
arguments=[15, 1, 0])
#arguments: mine the top 50 rules with confidence of 100% (1) for real news (1)
spmf = Spmf('TopSeqClassRules', input_filename='rules.txt',
output_filename='/path/to/file/realRules.txt',
spmf_bin_location_dir='/path/to/file/',
arguments=[50, 1, 1])
#create a C-SVM model with a linear kernel
m = svm_train(y, x, '-s 0 -t 0')
```

Appendix F

NEURAL NETWORK ERROR EXAMPLES

#32 4月14日，美国小哥实拍纽约的即时街景，解说美国疫情现状，当地市民已经意识到口罩的重要性，很多人戴上口罩。此外，他还探访纽约的餐厅、咖啡馆、超市、医院、公立学校。。。。

On April 14, an American guy shot a real-time street view of New York to explain the current situation of the epidemic in the United States. Local citizens have realized the importance of masks and many people wear masks. In addition, he also visited restaurants, cafes, supermarkets, hospitals, public schools in New York...

#33【“噢该死，这说的是美国”！美脱口秀演员抖包袱式讽刺美国】“中国让50万国民死于新冠病毒，占了全球（新冠）死亡人数的20%.....啊抱歉，这说的是美国。”当地时间4月13日，美国脱口秀演员李·坎普在RT“今夜编辑”节目中上演了颇为讽刺的一幕：他先是各种“抖包袱”借新冠疫情、种族和人权问题等“狂批”中国，随后突然发现自己“看错剧本”：啊抱歉，这说的是美国.....

[“Oh damn it, this is about America”! U.S. talk show actors sarcastically satirize the U.S.] “China has allowed half a million people to die from the new coronavirus accounting for 20% of the global (COVID-19) death toll...Ah sorry, this refers to the United States.” On April 13th local time, American talk show actor Lee Camp performed a rather ironic scene on the RT “Redacted Tonight” program: He was first made all kinds of jokes using the COVID-19 epidemic, race and human rights issues to give strong criticism to China, then suddenly found out that he had “read the wrong script”: ah, sorry, this refers to the United States...

#48 据俄罗斯卫星通讯社报道：日本在调运中国捐赠的2000检测试剂盒中，用的是美国运输机，到了目的地后，日本工作人员怎么也找不到这批检测试剂盒，再三追问，美军官员表示，由于失误，导致这批物资错运到美国本土。日本要求美方归还物资，美共和党议员卢比奥表示：检测试剂盒已经用完了，讨论归还的问题意义已经不大，美日应该联手共同对抗疫情问题。

According to a report by the Russian Satellite News Agency: Japan used a U.S. transport plane to transport the 2000 test kits donated by China. After arriving at the destination, the Japanese staff could not find the test kits. After repeated questioning, the U.S. military official said that due to mistakes, these materials were transported to the United States by mistake. Japan requested the U.S. to return the supplies. The U.S. Republican Congressman Rubio said the testing kits have been used up, and it is not meaningful to discuss the return. The U.S. and Japan should join forces to fight the epidemic.

#118 张文宏医生自爆收入和全家福，1969年出生，浙江瑞安人，从小喜欢读书，学习成绩优秀，在老家被邻里称为神童。张医生高考考入上海医学院，八年硕博连读后留学。现是全球知名的传染病专家。自爆年收入184万，来源于3个方面: 1.华山医院呼吸科主任，工资奖金年入50万。2.国家重点项目学科带头人，年入120万。3.每年在全球顶尖医学杂志（柳叶刀）上发表一篇署名论文，专利费2万美元。

Dr. Zhang Wenhong publicly disclosed his income and family situation. He was born in Ruian, Zhejiang Province in 1969. He liked reading since he was a child. Due to his excellent academic performance, he was called a child prodigy by his neighbors in his hometown. Dr. Zhang was admitted to Shanghai Medical College following the college entrance examination. He studied abroad after eight consecutive years of master's and doctoral studies. He is now a world-renowned infectious disease expert. His self-disclosed annual income of 1.84 million comes from three sources: 1. Director of the Department of Respiratory Medicine of Huashan Hospital, with an annual salary of 500,000. 2. Leader of national key projects, with an annual income of 1.2 million. 3. Published a signed paper in the world's top medical journal (The Lancet) every year, with royalties of US$20,000.