Computational Models of Interactional Stancetaking

Sumeet Kumar

Institute for Software Research
Carnegie Mellon University

This thesis proposal is submitted for the degree of
Doctor of Philosophy

School of Computer Science
January 2019
Abstract

People express their opinions on blogs and other social media platforms. Automated ways to categorize views of people in such user-generated corpora is of immense value. My thesis aims to develop computational models to learn to predict the stance of users who are active on social media platforms like Twitter.

In particular, I explore machine learning approaches for stance learning, which involves learning people’s opinion about a topic of interest. Most existing studies on stance learning take a simplistic view that assumes a ‘sentence’ (like a Tweet) holds a perspective that is independent of the context and the author. This approach to stance learning ignores the complex activities and interactions of social media users. According to one definition of stance, it is a public act of evaluating objects, positioning subjects, and aligning with other subjects. In the same spirit, I approach stance learning in the broader context of social action wherein authors take a stance to position themselves on topics of interests, thereby aligning with other stance takers. This approach, therefore, brings a new direction to the stance learning problem which is grounded in social theory and is more amenable to analyzing conversations on social media.

In this thesis, I plan to develop models for different interactions, and then find ways to combine them to improve the stance prediction accuracy. I would like to apply these models to quantify polarization in online communities.
# Table of contents

List of figures vii

List of tables ix

1 Introduction 1
   1.1 Background and Motivation .......................... 2
   1.2 Prior Work ........................................ 4
   1.3 Datasets ............................................ 8
      1.3.1 Existing Datasets ............................. 8
      1.3.2 New Datasets .................................. 10
   1.4 Summary of Proposed Work and Thesis Timeline .......... 10

2 Proposed Work 13
   2.1 Stance Learning from Text Data .......................... 13
      2.1.1 Related Work ................................... 15
      2.1.2 Methodology ..................................... 16
      2.1.3 Preliminary Results ............................ 16
   2.2 Person’s Stance from Multiple Posts .................... 19
      2.2.1 Proposed Methodology .......................... 21
   2.3 Improved Stance Learning by Leveraging Network Information 22
      2.3.1 Related Work ..................................... 22
      2.3.2 Proposed Methodology .......................... 23
   2.4 Learning Agreement/Disagreement Among Users from Online Conversations 25
      2.4.1 Related Work ..................................... 25
      2.4.2 Methodology ..................................... 26
      2.4.3 Preliminary Results ............................ 28
   2.5 Co-learning : Stance Learning from Multiple Modalities (text and networks) 30
   2.6 Application of Stance Mining in Social-Media Polarization .... 32
List of figures

1.1 Common approaches to stance learning ........................................... 3
1.2 The stance triangle ................................................................. 4
1.3 Proposed changes to the current stance learning paradigm ............... 5
1.4 Application of interactional stancetaking ..................................... 11
1.5 Chapters summary .................................................................. 12

2.1 Stance learning using social-media hashtags ................................. 14
2.2 Conditional LSTM ................................................................. 20
2.3 AttSum: Summarizing with attention .......................................... 21
2.4 Co-learning: A joint model to learn from text and graphs .............. 23
2.5 Stance in Twitter threads ......................................................... 26
2.6 LSTM for stance learning ......................................................... 27
2.7 Multi-modal LSTM for stance learning ....................................... 28
List of tables

1.1 Manually labeled datasets created in prior research .......................... 8
1.2 SemEval 2016 Stance Dataset .................................................... 9
1.3 Multi-stance dataset ................................................................. 9
1.4 Constance dataset ................................................................. 10
2.1 MAB Evaluation: F1 score for different algorithms ......................... 17
2.2 Results: Stance labels for reply posts ........................................ 29
Chapter 1

Introduction

People express their opinions on social media platforms. Automated ways to categorize views of people in such user generated corpora is of immense value. In particular, in polarized communities which are increasingly common these days, mining users opinion can enable situational awareness and help to bridge communities. To mine social-media for opinions, stance learning – which involves finding people’s opinion about a topic of interest – provides a natural way to model social interaction, therefore lending itself to study users’ opinions. So it’s not a surprise that stance learning has become an active area of research [28, 49, 34]. However, most existing study on stance takes a simplistic view that assumes a ‘sentence’ (like a Tweet) holds a perspective that is independent of the context and the author. This approach to stance learning ignores the complex activities and interactions of social media users. Bois et al. [16] (page 163) defines the act of stance as ‘a public act by a social actor, achieved dialogically through overt communicative means, of simultaneously evaluating objects, positioning subjects (self and others), and aligning with other subjects... ’. In the same spirit, I approach stance in a broader context of social action wherein authors interact on social-media to align themselves with other stance takers.

In this research, I plan to design models for different interactions and then combine them to improve the stance prediction accuracy. First, to predict stance from text, I propose a weakly supervised learning method that reduces the cost of collecting data to train models for new topics. In this approach, I use specific hashtags found on social media platforms that carry stance information. Using these hashtags as noisy labels, I build a classifier that only needs labeled data for validation, yet achieves accuracy comparable to the state-of-the-art on a human-labeled stance dataset. Second, I plan to extend the text-based stance learning to use multiple posts from a user. Using different social media posts of users is expected to improve the stance classification accuracy. Third, I plan to extract different types of networks on Twitter (e.g. follower network, mentions network), and use graph-based algorithms to align
users based on their similarity in the networks. Fourth, I want to use conversation threads to learn the authors’ position about an issue based on their posts and the replies to these posts. On Twitter, users can reveal their alignment with other users based on the stance they take in replying to other posts. Such interactions are useful in getting disagreement among users as most other interactions, e.g. liking and retweeting only reflect alignment. After developing independent models for different types of interactions, I intend to propose a joint model that combines the various sources of stance information. To combine different modalities of information (e.g., users’ follower graphs and users’ likes), I plan to embed features from all information sources into one continuous representation space. Finally, I would like to apply these models to measure community polarization.

Unlike prior work, our approach positions users as central and their alignment as a crucial to learning stance. This approach, thus, brings a new direction to the stance learning problem which is grounded in theory and is more amenable to conversations on social-media. This research can be applied to improving the understanding of community polarization and partisanship.

Next, I provide some theoretical background on stance taking, related prior work, the existing and the new datasets that I intend to use, and the time line of the proposed work. Details on the proposed work can be found in the second chapter.

1.1 Background and Motivation

People express their opinion on different topics on social media platforms like Twitter and Facebook. Algorithms have been developed to automatically extract peoples opinion form these vast corpora of user-generated data. In particular, sentiment mining has remained an extremely popular field of research in the last decade [46]. However, sentiment mining aggregates users’ opinions. In sentiment mining, we are interested in finding general perception of an entity. For example, the company ‘Apple’ may want to find about their brand perception i.e. they are generally discussed in a positive way or or a negative way. In such tasks, we first obtain a large set of data by searching the social media platforms for the topic and then obtain a sentiment score for each data item. Finally, we average the scores obtained to get the perception. This approach of analysis is beneficial in tasks like learning about a brand and measuring a product’s rating. However, there are other tasks in which, we are interested in information at the level of users. For example, we might be interested in finding the perception of people about a party or a political candidate. Such tasks are better approached by evaluating opinions at the level of users. This aspect is further highlighted when the goal is to understand the view of users engaged in a rumored discussion. In discussing such
1.1 Background and Motivation

In controversial topics, people take sides and are invested in the discussions at various levels. As described by Kiesling et al. [35], there are three dimensions to analyzing conversations on social media: a) Affect b) Alignment c) Investment. Sentiment mining stays at the level of affect. In stance learning, we consider affect and alignment. In conversations on social media, there is yet another aspect of ‘Investment’. In this work, I tend to discount ‘Investment’ as the prime application of this work is to understand polarization on social media about a controversial topic. When it comes to controversial topics, people take a side, i.e. either they are either pro or they are anti.

Fig. 1.1 Common approaches to stance learning. The figure illustrates that there are three common approaches to stance learning: a) stance from text (e.g. a twitter post) b) stance from multiple tweets c) stance on multiple topics from text.

Most existing work on stancetaking use text (e.g. from a Tweet) to predict the stance of the author of the text. I summarize the current approaches to stance learning in Fig. 1.1. As shown, there are three common approaches to stance learning: a) stance from a tweet b) stance from multiple tweets c) stance on multiple topics from text (more on this in the prior work section). All theses approaches assume that the given text contains enough information to predict stance of users. However, as shown by Joseph et al. in Constance [34], in addition to a tweet, providing more context relevant information (e.g. more relevant tweets) improves the agreement between annotators who are labeling stance of users. Thus, it is natural to expect that other relevant interactions on social media would also improve the stance prediction accuracy.

In this work, we focus in aligning users to better evaluate their stance. This approach closely follows the definition of Stance proposed by Prof. Du Bois [16], who argues that stancetaking is the public act of simultaneously evaluating objects, positioning subjects and aligning with other subjects. This definition is illustrated in a Fig. 1.2 which is described as
‘The stance triangle’. As shown in the figure, alignment (or dis-alignment) happens as users display similarity and difference with respect to their evaluations.

Fig. 1.2 The stance triangle, adapted from Du Bois [16], page 163.

There are many challenges in applying interactional stance taking to improve stance prediction by aligning users. The primary challenge is the multiple ways people can interact on Twitter. The popular way is to ‘tweet’ which is a way to broadcast opinion. However, in addition to tweeting, one can retweet posts, like posts, reply to a post, make connections (like become a follower) and more. All these types of interactions have relevance in finding users stance and their alignment. My goal, in this thesis research, is to explore which of these interactions are useful for learning stance and how the learners that are trained on one type of interaction can be combined with others.

As shown in Fig. 1.3, the current stance learning system ignore complex interactions. In this thesis, I intend to move from the current text based stance learning paradigm to a graph based stance learning approach that can use text as input, in addition to other kinds of Twitter interactions.

1.2 Prior Work

Prior work on Stance learning have appeared in primary two flavors a) Ideological leaning on social media platforms [49, 34, 60] b) Agree or disagree about a topic by taking a stand in debates [61, 28, 62]. I summarize recent contributions on both types of work next. In addition, I also discuss applications of stance learning in rumor identification and use of neural-networks for stance learning.
1.2 Prior Work

Stance learning from social-media data

Researchers have used social media as a source of stance data. Mohammad et al. [49] created a stance dataset using Tweets and explored a few algorithms to predict stance in those tweets. The dataset obtained from Twitter contained text from on five controversial topics and was used in a SemEval, 2016 competition. Many researchers used this dataset and tried many different types of algorithms to predict stance [4, 47, 72]. However, as reported in [49], none of them exceeded the performance achieved by a simple algorithm that uses word n-grams and word-embedding as features. Even neural-network models [3] that uses bi-directional conditional encoding did not perform better than simple n-grams based models. Researchers have also looked at different ways to train stance models. By analyzing temporal tweeting activity of politicians and the way issues can be framed on Twitter, [33] designed a weakly supervised model for learning stance. They suggested to use content, frames jointly, and temporal activity to build local models and combine them through Probabilistic Soft Logic.

Though not directly on stance learning, there is also a variety of research on signed networks [43]. Wang et al. proposed [70] SiNE to learn signed network embedding. SiNE learns a low dimensional representation of nodes while preserving the ideas of structural balance theory. In [73], authors build people to people network. Please check [65] for a survey on data mining approaches for signed networks. The work on the signed network could be useful in the chapter where we use different networks to infer stance.

Fig. 1.3 Proposed changes to current stance learning paradigm. The different types of arrows show different interactions. The ‘thumb up’ and ‘thumb down’ icons show the stance taken by users. The two blue ovals indicate topics of discussion.

1http://alt.qcri.org/semeval2016/task6/
Stance learning in debates

Earlier work on stance learning revolved around debates [61, 28, 62]. Using a manually annotated corpus, Somasundaran and Wiebe [61] constructed an arguing lexicon to recognize stance in online debates and used an SVM classifier resulting in above 60% accuracy. The authors used data on debate posts in four domains ‘Gun rights’, ‘Gay rights’, ‘Abortion’ and ‘Creationism’. Ozer et al. tried to cluster politically motivated users into communities. They used structural balance theory to build a user-connectivity network using endorsements and tried three non-negative matrix factorization approaches to detect communities. Tu et al. proposed context-aware embeddings that attempt to use semantic relationships between users to preserve the diverse roles of interacting users. Some researchers have also considered adding different types of constraints in models, e.g. [28] used author constraints (AC) in which two posts written by the same author for the same domain should have the same stance. In [27], the authors explored more constraints that include ideology constraints and user-interaction constraints. By modeling the problem as an optimization problem and solving it by integer linear programming (ILP), they report an improvement in accuracy in the range of 2-10%. Due to the short text and noisy nature of social media, in most cases, this line of research on debates can’t be directly applied to social media data.

Stance in rumor and misinformation identification

This task of stance learning got popularity when it was realized that stance could be used to identify fake news ² and rumors ³. Finding misinformation on social media platforms has been an active area of research in recent years. In this type of work, the researchers use the content of a post to determine the veracity typically uses language features. For example, Rashkin et al. [57] discuss language characteristics of real news compared to satire, hoaxes, and propaganda. They found that Linguistic Inquiry and Word Count (LIWC) and sentiment lexicon features are useful to understand the differences between fake and more reliable digital news sources. This can be further extended by including information present in more reputable sources like using knowledge graphs to find factual discrepancies [29]. When network features are available (e.g. when an article is posted on social media), researchers have shown that including social-network features in addition to content features, outperform lexical based models [69].

There is another line of research on rumor detection that uses stance in the reply posts. This kind of work was in initiated by the Pheme project ⁴ and was popularized in a SemEval

²http://www.fakenewschallenge.org/
³http://alt.qcri.org/semeval2017/task8/
⁴https://www.pheme.eu/
1.2 Prior Work

2018 competition (task 8) \(^5\). The task involves predicting stance (‘supporting’, ‘denying’, ‘commenting’ and ‘querying’) in replies to rumor posts on Twitter and is described in zubiaga2015crowdsourcing and zubiaga2016analysing. A number of researchers used this dataset and proposed many algorithms. For this dataset, Derczynski et al. proposed an LSTM that uses branch of conversation trees to classify stance in reply posts, and zubiaga2018discourse used and compared many different sequential classifiers. My work extends this thread of research by using different ways to learn stance from social media conversations.

**LSTM and multi-modal neural networks for stance learning**

Deep neural networks (DNN) have shown great success in many fields [30]. Researchers have used DNNs for various NLP tasks like POS tagging, named entity recognition [10]. Among the tasks related to stance learning, sentiment detection tasks are of great interest. Tang et al. [64] used gated recurrent neural network (GRNN) for sentiment classification. He showed that the GRNN model could dramatically outperform other standard recurrent neural networks (RNNs). In another recent work, Yoon Kim [37] used convolution neural networks (CNN) for various NLP tasks including sentiment analysis. They showed that a simple CNN with parameter tuning could provide state of the art results on sentiment analysis.

In DNNs, our prime interest is in multi-modal techniques [51] that can combine multiple types of features extracted from social media interactions. The embedding techniques used by [52] (for AVEC 2016 challenge) considered top audio and video features for fusion. They used a logical ‘AND’ based fusion technique which may not be the best way to jointly represent the modalities. I will also explore ways to fuse different sources of information.

On stance learning, [78] used an average of word vectors from each tweet as an input to their LSTM model. They trained their LSTM model using branches of twitter conversations where mean-of-words is used as input and stance label is used as output. Sequential classifiers like LSTMs are biased to use prior inputs to predict outputs. For example, an LSTM can be used to predict the next word for the partial sentence ‘The flower is __’, given a large amount of training data the model estimate that ‘beautiful’ or ‘fresh’ is more likely to the last word. However, when it comes to tasks like stance classification in threaded discussions, each reply is made against another post (and not necessarily to the series of earlier posts). Does the sequential nature of responses make it more suitable for an LSTM like model? Or would a model that can learn the differences between the source and reply tweets are more ideal for stance classification? These are some of questions that we will investigate in this research.

In most research described above that used Social-media data, the stancetaker (as in author of text) are not explicitly considered, and aligning authors are not an important

---

\(^5\)http://www.aclweb.org/anthology/S17-2006
goal. However, in many applied areas where authors take clear stance (e.g. in polarized communities), authors alignment is important as asserted in [16]. Our research explores this critical aspect of stance learning.

1.3 Datasets

There are some datasets on stance learning that were built in prior research by other researchers which I intend to use in my research. In addition to the existing datasets, I am currently building a dataset that has stance labels for replies in conversation threads. I also plan to develop another dataset that has Twitter users’ stance labels for multiple controversial topics using the users’ timeline (all tweets). I describe both the existing and the new datasets next.

1.3.1 Existing Datasets

As explained in Fig. 1.1, existing approaches on stance learning that uses Twitter data can be categorized as one of the three types a) Single Tweet Single Target (STST) b) Single Tweet Multiple Targets (STMT) and c) Multiple Tweets Single Target (MTST). We summarize the three existing datasets in Table 1.1.

<table>
<thead>
<tr>
<th>Reference</th>
<th>Type</th>
<th>Targets/Topics</th>
<th>Count (total/train/dev/test)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SemEval 2016 [49]</td>
<td>STST</td>
<td>Atheism, Climate Change is concern, Feminist Movement, Hillary Clinton, Legalization of Abortion</td>
<td>4163/ 2914/ NA/ 1249</td>
</tr>
<tr>
<td>MultiStance [60]</td>
<td>STMT</td>
<td>Donald Trump, Hillary Clinton, Bernie Sanders, Ted Cruz</td>
<td>4455/ 3119/ 446/ 890</td>
</tr>
<tr>
<td>Constance [34]</td>
<td>MTST</td>
<td>Donald Trump, Hillary Clinton, Neutral</td>
<td>1130/ 562/ 250/ 318</td>
</tr>
</tbody>
</table>

Out of these three datasets, SemEval 2016 [49] contains text and stance labels for five topics (see table. 1.2). Because this dataset does not share the users information, we use this dataset only for the first chapter of this thesis that uses text for stance labeling. The other
two datasets (created in [34] and [60]) contain id of the tweets in addition to the text. Using
tweet ids, we can retrieve users, thus, these two datasets can be used to create a users-stance
dataset. We intend to use this users-stance in the chapters that are on aligning users.

**SemEval 2016 Stance Dataset (Single Text Single Target)**

In this section, we describe a human-labeled dataset that was built in prior research [49] and
was used for a competition in SemEval 2016. The dataset is summarized in Tab. 1.2.

<table>
<thead>
<tr>
<th>Topic</th>
<th>Train</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atheism</td>
<td>513</td>
<td>220</td>
</tr>
<tr>
<td>Climate Change</td>
<td>395</td>
<td>169</td>
</tr>
<tr>
<td>Feminist Movement</td>
<td>664</td>
<td>285</td>
</tr>
<tr>
<td>Hillary Clinton</td>
<td>689</td>
<td>295</td>
</tr>
<tr>
<td>Legality of Abortion</td>
<td>653</td>
<td>280</td>
</tr>
</tbody>
</table>

This dataset, which I think is the most popular dataset for stance learning, has text data
on five topics that are ‘Atheism’, ‘Climate Change’, ‘Feminist Movement’, ‘Hillary Clinton’
and ‘Legality of Abortion’. This dataset has no development/validation set.

**Multi-target Stance Dataset (Single Tweet Multiple Targets)**

Sobhani et al. built a dataset that contains stance labels for multiple target from single tweets.
This dataset is summarized in Tab. 1.3. The dataset has stance labels for two targets given
one tweet. The target pairs are ‘Clinton-Sanders’, ‘Clinton-Donald Trump’ and ‘Cruz-Donald Trump’.

<table>
<thead>
<tr>
<th>Target Pair</th>
<th>Train</th>
<th>Dev</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clinton-Sanders</td>
<td>957</td>
<td>137</td>
<td>272</td>
</tr>
<tr>
<td>Clinton-Donald Trump</td>
<td>1240</td>
<td>177</td>
<td>355</td>
</tr>
<tr>
<td>Cruz-Donald Trump</td>
<td>922</td>
<td>132</td>
<td>263</td>
</tr>
<tr>
<td>Total</td>
<td>3119</td>
<td>446</td>
<td>890</td>
</tr>
</tbody>
</table>
Human Labeled Multi-tweet Stance Dataset (Multiple Tweet Single Target)

Joseph et al. built a dataset that has stance labels for 1130 users for ‘Support Trump- Oppose Clinton’, ‘Neutral - I don’t know’ and ‘Oppose Trump - Support Clinton’. The dataset is summarized in Tab. 1.4.

Table 1.4 Constance Dataset: Human labeled that uses multi-tweets to determine stance of a Single target [34]

<table>
<thead>
<tr>
<th>Target</th>
<th>Train</th>
<th>Dev</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oppose Trump- Support Clinton</td>
<td>562</td>
<td>250</td>
<td>318</td>
</tr>
<tr>
<td>Support Trump- Oppose Clinton</td>
<td>562</td>
<td>250</td>
<td>318</td>
</tr>
<tr>
<td>Neutral</td>
<td>562</td>
<td>250</td>
<td>318</td>
</tr>
</tbody>
</table>

Dataset on stance labels for replies in rumour threads

This dataset was created as a part of the Pheme project which aims to find and verify rumors shared on social-media platforms [79, 80]. The dataset consists of Twitter conversation threads on nine different events and contains three types of annotations. Each thread is labeled as either rumor or non-rumor. Rumors are annotated as true, false or unverified. For a subset of the true rumors, we also have stance labels for each reply in the threaded conversations. The stance labels are ‘supporting’, ‘denying’, ‘commenting’ and ‘querying’.

1.3.2 New Datasets

In addition to using the existing datasets, I am developing two dataset for this thesis. The first new human-labeled dataset, being built using the help of MTurk workers, is composed of stance labels of replies to posts in controversial threads. The other dataset that I am planning to build will have stance labels for many users on a few controversial topics.

1.4 Summary of Proposed Work and Thesis Timeline

I divide this thesis in seven chapters (see Fig. 1.5).

1. Introduction
1.4 Summary of Proposed Work and Thesis Timeline

Fig. 1.4 Application of interactional stancetaking to understand community polarization.

2. Stance learning from text data

3. Person’s stance from multiple posts

4. Improved Stance learning by leveraging network information

5. Learning agreement/disagreement among users from online conversations

6. Co-learning : Stance learning from multiple modalities (text and networks)

7. Application of stance mining in social-media polarization and echo-chambers

Chapter one introduces this thesis work.

In the second chapter, I explore various ways to extract stance from text. In particular, I find an inexpensive way to build datasets for new topics that can be used to train models for stance learning.

In the third chapter, I use multiple posts by a user to get a better estimate of his or her stance on different topics. I start with simple models that use stance averaging, but will also try more complex neural-network based models like conditional LSTMs and attention mechanism to combine information in multiple posts.

The fourth chapter extends the third chapter by using user’s network information for stance learning. This can potentially be done by learning continuous representation of users using their activity and network. Such representations help to include users’ information in other learning models (e.g. text based stance learning model).

In the fifth chapter, I intend to use conversations on social media platforms as a way to align users. I am particularly interested in posts on controversial topics and replies to those posts where other users express their agreement and disagreement.
In the sixth chapter, I plan to combine and extend the different methods of stance learning that I have developed in the first four chapters. On Twitter, stance can be observed in users’ posts but can also be derived from ‘likes’, ‘quotes’, and ‘retweets’. Besides, because of homophily effect, users follower-followee relationships can be used to predict stance. This chapter combines the multiple ways to infer stance to improve the stance prediction accuracy.

In the seventh chapter, I intend to use the stance models learned in earlier chapters and apply them to datasets on controversial topics (see Fig. 1.4). As controversial issues often tend to have polarized discussions, I hope to find and quantify polarization in online communities.

I plan to finish the thesis by following the below time line:
Chapter 1 and Chapter 2 - Completed
Chapter 5 - In progress.
Chapter 3 - Plan to complete in Spring 2019
Chapter 4 and Chapter 6: Summer 2019
Chapter 7: Fall 2019

Fig. 1.5 Summary of the Thesis Chapters. The different colors of lines show different interactions. Solid lines indicate explicit relationships (like follower-followee) and dashed lines indicate inferred relationships.
Chapter 2

Proposed Work

I divide the planned research in this thesis into the following six topics.

1. Stance learning from text data
2. Person’s stance from multiple posts
3. Improved Stance learning by leveraging network information
4. Learning agreement/disagreement among users from online conversations
5. Co-learning : Stance learning from multiple modalities (text and networks)
6. Application of stance mining in social-media polarization

Topic 1, Topic 3, and Topic 4 are independent of each other. Topic 2 depends on topic 1. Topic 5 and Topic 6 depend on all previous topics. Next, I describe the high-level ideas in these topics. For Topic 1 and Topic 4, I also present the preliminary results.

2.1 Stance Learning from Text Data

Researchers have explored ways to extract stance from text data [49, 34, 60]. Table 1.1 summarizes three datasets on stance learning that is based on text data. Such manually labeled datasets are of great value as it allows to compare various learning algorithms with respect to humans expectations. However, in the real-world, models built on these small datasets do not generalize, especially to unseen data. Also, these hand labeled datasets are only available for a few topics, and models trained on these datasets are not appropriate for use with new topics. In this research, we propose a more flexible approach to build datasets
Proposed Work

Fig. 2.1 Stance learning work-flow: Given a topic, using sentiment analysis we first extract the top hashtags having ‘pro’ and ‘anti’ stance on that topic. Using these stance hashtags, we download more data that are available on Twitter. This large dataset is then used to train neural-network models for stance classification. Moreover, the goal to pick the best pro and anti hashtag combination for classification can be modeled as a multi-armed-bandit (MAB) problem. Different optimization strategies are used to pick the best pro and anti hashtag to train stance learning models for classification. This approach only requires a few hand-labeled examples for validation.

The use of hashtags on social-media is ubiquitous and the tweets associated with some of these hashtags could provide a useful signal for weak supervision. Prior research have explored using hashtags for emotion [12, 56] and topic [71] labeling tasks. In this research, we propose to use hashtags for stance learning. Hashtags appended at the end of the tweet in social-media posts tend to carry stance [17]. Searching for such hashtags, on Twitter, we observed that certain hashtags carry stance information (e.g. ‘#climatehoax’). In this research, we propose an approach that uses data obtained by searching hashtags to train deep-learning models that can predict stance in a text (see Fig. 2.1). We use text sentiment to identify potential hashtags that can carry stance information. Given a topic, we pick a few hashtags that have either higher mean positive sentiment or mean negative sentiment. We call these hashtags ‘stance-tags’ as these hashtags can carry stance information. Then, we collect a large number of tweets using these stance-tags. Using weak supervision from the stance of stance-tags, we train two deep-learning models to predict stance in the text that don’t necessarily contain hashtags. We observe that depending on hashtags and the timing of data collection, the accuracy of stance learning could fluctuate. The fluctuation is because of the noisy nature of social-media where hashtags could be used for multiple purposes. To
address this issue, given a few labeled examples, we use the multi-armed-bandit (MAB) optimization to find stance-tags that are most suitable for stance learning in the long run. We validate our approach with experiments on a hand-labeled dataset with five different topics, obtaining results comparable to the state-of-the-art. To the best of our knowledge, this is the first work that suggests a principled approach for finding and using hashtags as labels for stance learning.

2.1.1 Related Work

Weakly Supervised Machine Learning

In the text domain, weakly-supervised training is common. Go et al. [24] used positive and negative smiley emoticons to train a sentiment classifier. Researchers have also tried hashtags for weak-supervision. Mahajan et al. [48] recently used hashtags posted with images for object detection. In the text analysis, hashtags have been used in sub-event discovery [76] and aspect-based opinion mining [45].

Though weak labels and unsupervised learning are common in many areas of text mining (like sentiment mining), for stance-learning most known work use labeled datasets. Mohammad et al. [49] briefly discussed the idea of using unlabeled tweets for improving their classification results. They explored two methods: first using additional data for training and second using word-association as additional features. They found that using data from certain hashtags as additional training data, they could improve their f-score by up to 4 percentage points for one target, but minimal for others. In comparison to their work, the main focus of this work is build a larger weakly labeled dataset for stance learning.

Multi-Armed Bandit Problem

In simple terms, a multi-armed Bandit (MAB) problem involves allocating a fixed set of resources between competing choices to maximize the long-term gain [63] (chapter 2). Such problems often involve exploration vs exploitation trade-off [1]. For example, a gambler playing with a number of slot machines (‘one bandit with many arms’) has to decide which machines to play and how many times to maximize his gain. This problem appears in different forms like the contextual-bandit and the collaborative-bandit. In this research, we design our hashtags selection problem as a multi-armed bandit problem. In each iteration, a k-armed bandit has to choose between the arms to use. For this type of problem, researchers have explored many optimization approaches: e.g. epoch-greedy [42], upper-confidence-bound (UCB) values [2], softmax [63] (chapter 2) and Bayesian [59] optimization strategies.
2.1.2 Methodology

Our research can be divided in three parts:

1. Stance-Tag selection: We find potential hashtags that can be used for weak-supervision by using the mean sentiment in text associated with a hashtag. By taking a top few positive and a top few negative sentiment hashtags, this step results in a list of pro hashtags and anti hashtags. We call these hashtags stance-tags.

2. Stance classification using deep-learning models: Using data collected by searching the stance-tags on Twitter, we train two state-of-the-art neural-network architectures, a long-short term memory (LSTM) and a convolutional neural network (CNN), for stance classification. We evaluate the trained models by verifying their performance on an existing human-labeled dataset (see Tab. 1.2).

3. Multi-armed bandit problem for the optimal stance-tag selection: Because of inconsistency in results for models trained on data collected by searching stance-tags, we propose a strategy (see algorithm 1) to find the most suitable stance-tags for training the neural-network models. We use four optimization strategies (ε-greedy [42], UCB [2], Softmax [63] (chapter 2) and Bayesian bandit [59]) to search the space of hashtags to find the one that is the best for stance learning. This step leads to a better pro and anti stance-tag pair, which finally results in a better classification performance.

2.1.3 Preliminary Results

For all five topics in the dataset, we used the Double MAB optimization algorithm (see algorithm 1) to find the best stance-tag-pair. In the process of optimization, for each time step, we store the actions (choice of stance-tags), maximum arm probability (confidence in stance-tag selection) and regret. In Table. 2.1, we summarize the final results obtained by training models based on optimal stance-tag-pair. We find an improvement in the f1-score for two cases (‘Atheism’ and ‘Abortion’) compared to random choices for pro and anti stance-tags made in the second part. This is expected as we have an optimization routine to select the better stance-tag pairs. Compared to prior work, for ‘Hillary Clinton’ our algorithm is under-performing by a large margin, so we further analyzed it. We think this poor performance could be due to two reasons: a) The time difference in SemEval data collection (in 2016) and our data collection (in 2018) b) The noisy nature of political conversations on Twitter. For three topics, we are able to get better than the performance
Algorithm 1 Double MAB: Given the set of pro-stance tags and anti-stance tags, Pro-MAB finds the best pro-stance-tag and Anti-MAB finds the best anti-stance-tag.

Require: MAB \textit{mabPro}, MAB \textit{mabAnti}, proHashtags \textit{H}_p, antiHashtags \textit{h}_a, timesteps \textit{T}

0: for \(i = 1 \rightarrow T\) do
  # Get Pro-MAB strategy
  1: \textit{strategyPro} = \textit{mabPro}.strategy
  # Choose pro-hashtag
  2: \textit{h}_pi = \textit{H}_p[\textit{strategyPro}.choice]
  # Get Anti-MAB strategy
  3: \textit{strategyAnti} = \textit{mabAnti}.strategy
  # Choose anti-hashtag
  4: \textit{h}_ai = \textit{H}_a[\textit{strategyAnti}.choice]
  # Get f1 for the selected pro- and anti- hashtags
  5: \textit{f1}_validation = \text{CNNClassification(} \textit{h}_pi.data, \textit{h}_pi.labels, \textit{h}_ai.data, \textit{h}_ai.labels) \text{)}
  # Set MABs reward
  6: \textit{mabPro}.reward = \textit{f1}_validation
  7: \textit{mabAnti}.reward = \textit{f1}_validation
  # Estimate regret
  8: \textit{mabPro}.estimateRegret()
  9: \textit{mabAnti}.estimateRegret()
  # Calculate total regret
  10: \textit{totalRegret} = \textit{mabPro}.estimateRegret() + \textit{mabAnti}.estimateRegret()
end for

reported in [49]. Note that [49] algorithm has better performance that all seventeen teams that participated in SemEval 2016 on this task.

Table 2.1 Results. F1 score for different algorithms. Rf1 refers to [49]. The top performers for each topic is highlighted in bold font. Double-MAB is the algorithm that we proposed to select the best combination of stance-tags.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Atheism</th>
<th>Climate Change</th>
<th>Feminist Movement</th>
<th>Hillary Clinton</th>
<th>Legal. of Abortion</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. Rf1 n-grams</td>
<td>65.2</td>
<td>42.4</td>
<td>57.5</td>
<td>58.6</td>
<td>66.4</td>
</tr>
<tr>
<td>b. a + Sentiment</td>
<td>65.2</td>
<td>40.1</td>
<td>54.5</td>
<td>60.6</td>
<td>61.7</td>
</tr>
<tr>
<td>c. Rf1 Best</td>
<td>68.3</td>
<td>43.8</td>
<td>58.4</td>
<td>64.7</td>
<td>66.9</td>
</tr>
<tr>
<td>d. Double-MAB</td>
<td><strong>69.1</strong></td>
<td><strong>61.2</strong></td>
<td>53.4</td>
<td>49.8</td>
<td><strong>68.1</strong></td>
</tr>
</tbody>
</table>
2.2 Person’s Stance from Multiple Posts

Unlike sentiment which is short term, the stance is more stable. Therefore, multiple posts of a user on a topic are likely to carry the same stance. Joseph et al. explored this idea in Constance [34], in which the author used Twitter data to find the stance of a few users about two political candidates (Trump and Clinton). In the study, in addition to a targeted tweet that was meant to determine the users’ partisanship, the authors also used four additional tweets of each user. Out of these four, two tweets were on politics, and two were non-political tweets. The authors concluded that providing political tweets lead to the best inter-annotator agreement, therefore, suggesting that context relevant tweets are useful for humans to predict stance of users better. In this chapter, I intend to take advantage of this stable nature of stance by using user’s time-line (all posts) to better predict their stance on given a topic.

Given that humans are better able to predict users’ stance when given more contextual information, it is natural to expect that computational stance learning models can also leverage additional relevant information. However, the challenge lies in finding which tweets among the set of all tweets on a user’s time-line are contextually relevant. This filtration of relevant tweets can be approached in two ways: a) Build a classifier that also ranks the tweets. In this approach, I use a text-based stance learning model that gives an estimate of the stance in the range -1 to 1 (anti to pro). Then I remove all tweets that are near the center of the scale. For the tweets that are left after filtering, I aggregate the stance prediction of all tweets that are relevant (towards the end i.e. -1 and +1). Thus in this approach, the stance classifier additionally acts as a filter. b) Build an explicit filter to remove non-relevant tweets. In this approach, a filtration model first ranks all tweets based on their relevance. However, it is challenging to build such a filtration model as a good filter will allow tweets that are useful to the stance learning model. Thus the filter model needs to have a prior understanding of the stance classifier that will be used after the filtration step. However, if we make certain assumptions, we can create simpler filters. For example, if we use a word or a set of words to indicate the topic of relevance, the filtering model could be a simple word search. This model would search for the terms in the tweets and will keep only the tweets that are relevant to the topic of stance. However, given a new topic, finding the set of terms for filtration is nontrivial. I, therefore intend to use the first approach.

There is not much prior work related to stance learning that explores relevancy of tweets as a part of stance classifier. However, in the field of information retrieval, many techniques are used to find relevant documents given a query. I want to borrow one such approach (called AttSum) that uses neural-networks with attention to find highly relevant sentences given a query [9]. I want to extend this technique with new sentence encoding model (Conditional LSTM based text encoding [3]) which is more relevant for the stance learning problem.
Next, I describe AttnSum and Conditional LSTM before I discuss my proposed approach.

Fig. 2.2 Bidirectional encoding of a tweet conditioned on bidirectional encoding of a given target ‘Legalization of Abortion’. Adapted from Augenstein et al. [3], page 878. \( h_4 \) and \( h_9 \) vectors are used to predict stance.

**Conditional LSTMs**

Augenstein et al. [3] proposed conditional LSTM that encodes tweets conditioned on the target (or topic) of stance. This model is used in an unsupervised way to train stance learner when, during the test, the target of stance is not made available. Therefore, the model needs to learn target agnostic features during the training process. As shown in Fig. 2.2, Augenstein et al. proposed an LSTM that use the target along with the tweet to generate what they call conditional encoding. As they discuss in the paper, using embedding of tweets and the target independently is not useful. Their best performing model used the bi-directional encoding of tweet conditioned on a bidirectional encoding of the target. The authors used two hidden vectors, one obtained at the end of the encoding process and on at the end of target encoding to build a classifier.

**Extraction based summarizing and filtering**

Cao et al. [9] introduced AttnSum which jointly learns distributed representation of text and attention weights for a given set of sentences and a query. The model in shown in Fig. 2.3.
2.2 Person’s Stance from Multiple Posts

2.2.1 Proposed Methodology

For stance learning given a topic and series of tweets from a user, I plan to combine AttnSum and conditional LSTMs. Conditional LSTMs, as described before, are good at generating sentence embeddings that are relevant for a given topic. If I use this encoding as an input to the AttSum model that uses text encoding to return relevant tweets given a query (in our case the stance proposition), I should be able to get the benefit of both the models.
2.3 Improved Stance Learning by Leveraging Network Information

The focus of this chapter is to leverage multiple networks to improve stance learning. Using Twitter data, multiple networks can be created using explicit links like the ‘following’ relationship in which one user follows other users, and implicit links like a user liking a post of another user or two users using the same hashtag. In this chapter, I propose a generic approach that allows to combine multiple networks and use these networks for users’ stance classification. In this approach, I learn the representation (embedding) on each node by using information from all networks. In the rest of this chapter, I assume to have two networks, one explicit (e.g. follower graph) and one implicit (e.g. user co-hashtag network). However, the proposed approach can be extended to any number of networks.

Let’s define the problem statement in a formal way: Let $G(V, E, A)$ be a network where $v \in V$ are nodes (or users), $e \in E$ are edges and $A$ is adjacency matrix. Let $F : v \rightarrow \mathbb{R}^d$ be the function that learns a $d$ dimensional representations ($z$) of a node. Let $Y$ be a matrix of user-hashtags usage that contains information of hashtags usage for each user, where $Y_{ij}$ indicates count of using $j$th hashtag by $v_i$th user. The goal of the algorithm is to learn $z_i$, a low-dimensional representation of user $v_i$ ($d << |V|$) that considers similarity in $Y$.

We want to encode users in a continuous space as a vector such that two similar users have a smaller vector distance compared to two users who are less similar. For example, we expect that users that use similar hashtags to stay be closer in latent representation space. Once, we create such node representation, classifying node can be trivially done a using a classifier that takes a node representation as input and outputs a stance label.

2.3.1 Related Work

Node representation (also called embedding) involves learning low dimensional vector encoding of nodes that maps local network structural characteristics to a continuous space representation space. Because these representations are useful in many tasks including node classification and link prediction, learning representation of nodes in networks is an active area of research. Though many algorithms have been proposed to learn representation of nodes on simple graphs [55, 66, 25], most research have only explored explicit unsigned edges. Very few work have explored node representation for signed networks, e.g. Kim et al. [36] used sign and direction information in edges to learn node representations that encodes structural information. However, even in in these work signs of edges that are explicit are known a priori. In contrast, in our research on Twitter users, edges like follower-followee
relationships only provide partial information as often it’s the interaction (e.g. liking, quoting, commenting and retweeting) that convey similarity in users’ thoughts. Thus, a crucial aspect of our work is the way we build users-networks from Twitter data.

Like learning node representation from graphs, graph based neural network that allows to directly exploit structured relationships in data is an active area of research. Defferrard et al. [13] used fast localized spectral filtering in convolutional-neural networks. Li et al. [44] used gates in graph-structured neural networks. More recently, [23] proposed neural message passing architecture that unifies many prior work on graph neural networks. We will also explore one of these architectures for learning representation of twitter users.

### 2.3.2 Proposed Methodology

Random walks on links in a network can be used to generate contexts that consist of proximity nodes. Like in language models, such context is then used to build embedding vectors (representations) often by training a shallow neural network. Learning low dimensional representation of nodes in networks allows mapping local structural characteristics to a continuous space representation. Learning these representations of nodes have helped to improve performance in many tasks including node classification and link prediction. Though models proposed in [55, 66, 25] learn good representation on simple graphs, they mostly explore binary edges, so are best suited for social-networks that have clear links as in friendship and follower-followee relationship. In this research, we propose to try an extension of Deepwalk [55] which has consistently performed well on many problem domains.

![Fig. 2.4](image)

Fig. 2.4 Most algorithms that learn node (user) representations ignore social media activity. We use social-media activities like tags to create a user’s preference vector. Preference vectors of different users are then compared using similarity measures to find similar scores. The similarity scores between users are used to create weighted virtual links, which are later used to learn better node representations.

We propose a model that allows to learn representation of users that preserves their similarity as evident from their attributes and preferences, and can be observed from social
media activities (see Fig. 2.4 for illustration). The approach in Deepwalk used explicit connections and ignored the activity of users. In this work, we extend the explicit links with the idea of virtual links. Virtual links consider users’ activities in creating new connections that don’t exist explicitly. Thus virtual links enable our model to learn representations based on users’ activities like hashtags usage and likes on social media platforms in addition to their explicit relationship with other users (see Fig. 2.4). The model also incorporates the idea that not all links are not equal. Virtual links are weighted connections based on some similarity metric and are learned in a way that preserves their interaction strength, and hence, are more practical.

After we enhance the explicit network with virtual links, the generation of node representation can be trivially done by using the Deepwalk approach. In Deepwalk, the authors used random walks to generate node contexts and train a neural network to predict neighbour nodes. I refer the reader to my paper [40] for details. In addition to this random walk approach, I would also like to try the recent improvements in the graph based neural networks for node representation learning. See [23] for details.
2.4 Learning Agreement/Disagreement Among Users from Online Conversations

In this chapter, I intend to use conversations on social media platforms as a way to align users. To find if a user agrees or disagrees using a response to a post, I am creating a human-labeled dataset as discussed in the dataset section in the first chapter. However, before this new dataset is ready, I have experimented with an existing datasets [79] that has stance labels for replies (categorized as ‘supporting’, ‘denying’, ‘commenting’ and ‘querying’ as shown in Fig. 2.5) for some rumorous conversation threads on Twitter. In this research, I extend the prior approach of using branches of conversation threads for training an LSTM [78]. I highlight the use and effect of different ways to encode text. In particular, I show that extracting emotions from the posts are very useful for learning the stance in the reply posts.

Intuitively, we expect that replies that deny a post tend to have more negative sentiment than replies that support the post. Thus emotion features in replies could be a useful predictor of stance. However, most existing research on predicting stance in replies uses text-pattern as the primary signal (see related work below). It is possible that some aspects of emotion could be directly captured while using text pattern. However, as we show in this research, an explicit use of emotion features performs far better than letting the model extract emotions from text. Our stance learning approach is based on a multi-modal Long Short Term Memory (LSTM) [31] model that combines text and emotions features extracted from the source and the reply posts to learn stance. To evaluate our models, we use a human-labeled Twitter stance dataset that contains stance labels for over two thousand rumourous threads related to eight different events. Our proposed model achieves the state-of-the-art performance on stance learning, out performing the last best known model by around 10% on f1-score (macro).

This research is divided in three parts. In the first part, we use a text based stance learning model that explores different sentence encoding choices. In the second part, I use emotion features based stance learning model that uses emotions extracted from text. In the final part, I describe a few variants of multi-modal neural-networks that combines text and emotion features and enables us to achieve the state-of-the-art performance in stance classification.

2.4.1 Related Work

There is a number of research work on rumor detection that uses stance in the reply posts. This kind of work was in initiated by the Pheme project \(^1\) and was popularized as a SemEval

\(^1\)https://www.pheme.eu/
Fig. 2.5 Twitter threads with stance and rumor labels. The conversation shown above has two branches a) T1–R1–R11 and b) T1-R2. R1 and R2 are 1st level reply tweets and R11 is a 2nd level reply tweet. Stance labels for each reply is relative to the tweet it is replied to i.e. stance for R11 is with-respect-to R1. In this research, we use models trained on such twitter conversations to predict stance labels for unseen twitter conversations.

2018 task 8 2. The task involved predicting stance (‘supporting’, ‘denying’, ‘commenting’ and ‘querying’) in replies to rumor posts on Twitter and is described in [79, 80]. A number of researchers used this dataset and proposed many algorithms. For example, [14] proposed an LSTM that uses branch of conversation trees to classify stance in reply posts, and [78] used and compared a number of different sequential classifiers. This work extends this thread of research by showing that emotions features are important for stance learning and rumor detection.

2.4.2 Methodology

There are a number of ways different features can be combined in a sequential learning process. These generally follow two patterns: a) Early fusion b) Late fusion [5]. We try both the approaches.

In early fusion, the different features are merged before the learning model uses them. In this model, a Recurrent Neural Network (RNN) uses features that are concatenated first. If E1, E3 and E3 are sentence embeddings and M1, M2, M3 are emotion encodings, the contented features (E1+ M1, E2 + M2, and E3+ M3) serves as an input to the RNN and hence at each time step, the RNN uses a text as well as emotion input, to output a stance label.

2http://www.aclweb.org/anthology/S17-2006
2.4 Learning Agreement/Disagreement Among Users from Online Conversations

Fig. 2.6 Recurrent Neural Network (RNN) architecture for stance labeling. E1, E3, and E3 are sentence embeddings and M1, M2, M3 are emotion encodings. At each time step, the LSTM uses a sentence embedding vector and an emotion embedding vector. The outputs of the two recurrent networks are added together to create a joint vector. This joint vector is passed through another single fully connected layer and a softmax layer to output a stance label.

In late fusion, the different features are first passed through a learning model and the outputs are combined to get a label, or the outputs are again passed through another learning model to get a label. As we show in Fig. 2.6, if E1, E3, and E3 are sentence embeddings and M1, M2, M3 are emotion encodings, at each time step, the first LSTM uses the sentence embedding vector (E1, E2, E3) and the second LSTM uses the emotion embedding vector (M1, M2, M3). The outputs of the two recurrent networks are fused together to create a joint vector. This joint vector is then passed through another fully connected layer and a softmax layer at the end to output a stance label.

In addition to early and late fusion approaches, there are ways to combine features in a hybrid way. For example, if one of the features set do not depend on prior or later features (i.e. independent of sequence), we can fuse the independent feature without passing through a sequence learner. We expect this to be particularly beneficial for the emotion features, so we designed a model that does not use a sequence model for emotion features. As shown in Fig. 2.7, if E1, E3, and E3 are sentence embeddings and M1, M2, M3 are emotion encodings, at each time step, the LSTM uses a sentence embedding vector and an emotion embedding vector. However, the emotion vector is added to the output of the recurrent network without passing through any sequence model. This joint vector is passed through a fully connected layer neural network and a softmax layer to output a stance label.

In addition to LSTMs that are good at capturing pattern in recurrence, we also try a convolution neural network (CNN) architecture. CNNs are commonly used in computer vision applications but are also getting popular in natural language processing. CNN are particularly useful for capturing local patterns. Since a source tweet and a reply tend to
local patterns as well, CNNs are also expected to be useful for capturing stance relationships. Our CNN architecture is a joint model that has two convolution unit, one for each type of encodings. The output of convolution operators are concatenated and the joint vector is passed through a fully connected layer and a softmax layer to output a stance label.

2.4.3 Preliminary Results

After basic cleaning of tweets text, we extract the necessary features which includes mean on Glove vectors [54], Skipthoughts [38], Deepmoji [18] and Bert [15]. These features are then used to train individual feature models as well as joint and hybrid models. In all cases we use the output of the last softmax layer as stance label. Following the tradition in prior work, we use event wise cross-validation which means out of eight events, seven events are used to train a model and the eighth event is used to validate the performance. The validation performance in the best F1-score obtained during the training epochs. This step is performed for all eight events, and mean of F1-score from all eight events are used to compare the models.

Our LSTM [31] models are built using PyTorch library \(^3\), and used feature vectors as input, adds an LSTM layer, a linear dense activation layer followed by a dropout (0.3) and uses softmax layer for output. The models are trained using Stochastic Gradient Descent (SGD) optimization with a cross-entropy loss function. The learning rate were tried from \(10^{-3}\) to \(10^{-1}\). We tried LSTM layer size from 16 to 256. We found that 16 or 32 size LSTMs are sufficient for this task.

\(^3\)https://pytorch.org/
The joint and hybrid models also followed the same architecture except the concatenation of features at various stages as explained in the methodology. The joint CNN model used kernels on size 2 (as it word on two embedding vectors at time) and stride 1. The learning rate was tested from 0.001 to 0.1, and .008 was found to work well. We again used Stochastic Gradient Descent (SGD) optimization with a cross-entropy loss function.

In Tab. 2.2, we show the results of the top performing models for each type of model along with their comparison with the prior work that got the best performance. As we can observe in Tab. 2.2, the joint model that uses skipthought and deepmoji vectors outperform all other models. The joint model of Bert, skipthought and deepmoji is slightly worse. The joint model on skipthought and Bert is almost equally good. However given that Bert requires an additional step of tuning and is larger (in size) than deepmoji vector, we recommend the skip-thoughts and deepmoji combination for this stance learning task.

Table 2.2 F1 score (macro) for different events. For each event code, data from the rest of events were used to train the model. The event code data was used for testing. This follows the tradition of prior research for an easy comparison. The F1 score is reported for the best performing classifiers.

<table>
<thead>
<tr>
<th>Model, Event →</th>
<th>CH</th>
<th>SS</th>
<th>FG</th>
<th>OS</th>
<th>GC</th>
<th>PM</th>
<th>PT</th>
<th>EE</th>
<th>Mean F1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LSTM Joint Models</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bert + DeepMoji</td>
<td>0.394</td>
<td>0.393</td>
<td>0.371</td>
<td>0.421</td>
<td>0.443</td>
<td>0.451</td>
<td>0.443</td>
<td>0.753</td>
<td>0.459</td>
</tr>
<tr>
<td>Skipthought + DeepMoji</td>
<td>0.439</td>
<td>0.438</td>
<td>0.391</td>
<td>0.472</td>
<td>0.501</td>
<td><strong>0.576</strong></td>
<td><strong>0.565</strong></td>
<td>0.638</td>
<td>0.502</td>
</tr>
<tr>
<td>Skipthought + Bert</td>
<td>0.456</td>
<td>0.461</td>
<td>0.415</td>
<td>0.457</td>
<td>0.523</td>
<td>0.484</td>
<td>0.501</td>
<td>0.714</td>
<td>0.501</td>
</tr>
<tr>
<td>Skipthought + Bert + Emotions</td>
<td>0.44</td>
<td>0.435</td>
<td>0.404</td>
<td>0.456</td>
<td>0.447</td>
<td>0.471</td>
<td>0.520</td>
<td><strong>0.762</strong></td>
<td>0.493</td>
</tr>
<tr>
<td><strong>CNN Joint Models</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Skipthought + DeepMoji</td>
<td><strong>0.463</strong></td>
<td><strong>0.481</strong></td>
<td><strong>0.448</strong></td>
<td><strong>0.499</strong></td>
<td>0.502</td>
<td>0.514</td>
<td>0.536</td>
<td>0.741</td>
<td><strong>0.526</strong></td>
</tr>
<tr>
<td><strong>Prior Research</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best Model [78]</td>
<td>0.465</td>
<td>0.446</td>
<td>0.373</td>
<td>0.475</td>
<td><strong>0.543</strong></td>
<td>0.457</td>
<td>0.379</td>
<td>0.657</td>
<td>0.474</td>
</tr>
</tbody>
</table>
2.5 Co-learning: Stance Learning from Multiple Modalities (text and networks)

Most prior work on stance classification that used Twitter data used users’ posts to learn stance. However, stance can also be learned from ‘likes’, ‘quotes’, ‘replies’ and ‘retweets’. Besides, because of the homophily effect, users follower-follower relationships can also be used to predict stance. In this chapter, I plan to use all different sources of information. To use multiple information sources, we can either combine the data or we can fuse the trained models.

To combine the data that are present in different formats (i.e. text and graph), a common approach is to convert all the data in any one of the formats. Thus, for stance learning, we can encode all posts (text) and graphs in a continuous embedding space. The other way is to convert all text-based inputs to graphs, thus converting all data formats into graphs. Since it's unclear the right way to convert text to graphs that preserves all information in the text, I plan to use the other approach of converting all inputs into an embedding space. For converting text to embedding space, I will try the multiple ways to encode sentences (like [38] or I will use the last layer of the neural text-based neural network classifiers. To convert graphs to a continuous representation space, a number of techniques have been proposed recently. Perozzi et al. [55] introduced DeepWalk, a model to learn latent continuous vector representations for nodes that encode social relations by using short random walks. They demonstrated the benefits for such representations for network classifications and anomaly detection problems. Tang et al. [66] introduced LINE, embeddings for large scale networks that try to preserve both global and local structure. They proposed an edge sampling method that keeps first-order proximity as well as second-order proximity and avoids the issue of an explosion of gradients in the stochastic gradient descent approach. Grover and Leskovec [25] proposed node2vec, an algorithm to learn continuous features representation for nodes using a biased random walk around the node. The learned features were applied to node classification and link prediction problems. In addition to random walks based approaches, there is a line of work on using linear algebraic formulation to learn a low-dimensional representation of matrix representing a network [7, 77, 67]. Recently, Huang et al. [32] introduced LANE that uses node labels in addition to geometric structure and attributes on nodes. I plan to try some of the approaches for in my work.

These are a number of ways to fuse outputs of multiple classifiers to improve the final classification accuracy. The most common way is to use a meta-classifier that works as an ensemble of classifiers [58]. Wozniak et al. [75] provides a good summary of the different ways to combine multiple classifiers.
Besides using the data and fusing the model, there is also an approach to improve the individual classifiers using the predictions of other classifiers. This approach is called co-training. In co-training [8], the different ways to infer stance serves as different views of data on which multiple learning algorithms are trained separately. Later, the predictions of each algorithm on new data are used to enlarge the training set of other algorithms. Thus, by co-training, I expect to achieve better accuracy in predicting users’ stance.
2.6 Application of Stance Mining in Social-Media Polarization

Community polarization and partisanship are heavily studied in social-science [11, 74]. Recently, with the presence of abundant data on social-media platforms, computer science researchers are exploring ways to utilize data to better understand such social interactions [22, 19, 21, 6]. Unlike most existing work on community polarization, I would like to use the stance models learned in earlier chapters to predict stance of users on controversial topics. Then I want to propose a metric that uses stance value of the users to quantify the extent of polarization in the community. As controversial issues often tend to have polarized discussions, I expect to find polarized community members in my datasets.

2.6.1 Related Work

Social media polarization have applied to political polarization (partisanship) and personal outlook (liberal vs conservative). Lahoti et al. [41] used constrained non-negative matrix factorization to explore liberal-conservative ideology space on Twitter. Using twitter dataset of controversial topics, the authors were able to separate users by ideology with over 90% accuracy.

Morales et al. [50] proposed a formula to measure polarization as a function of the difference in size between both populations $\Delta A$ and the poles distance $d$. The extent of polarization is defined as $\mu = (1 - \Delta A)d$, where $A$ stands for the area associated to each ideology, and $d$ stands for the pole distance. Using Twitter conversation about the late Venezuelan president, they found a good agreement between their results and the offline data.

Guerra et al. [26] performed a compared networks that arise in polarized non-polarized contexts to show that modularity (a common polarization metric) is not a direct measure of distance between polarized groups. They proposed a metric that is based on the boundary of potentially polarized communities. Their experiments showed that this new metric is better captures the notions of antagonism and polarization.

More recently, Garimella et al. [20] tried to quantify discussions on controversial topics on a graph-based three-stage pipeline. They first build a conversation graph about a topic which they partition to identify potential sides of the controversy. They measure the amount of controversy from a number of different characteristics of the graph. Their approach to quantify polarization used a modified version of random walks, which they claim is able to discriminate controversial topics with better accuracy than most existing work.
2.6.2 Proposed Methodology

None of the work mentioned above, use stance of users as the basis to quantify polarization. I propose to use stance value of individual users to quantify polarization. I would like to extend the prior work by evaluating the metric that fits in the framework of stance learning. In addition, I also want to try the feasibility of using Wasserstein distance [39] as measure of community polarization.
References


