USING MODULAR ARCHITECTURES TO PREDICT CHANGE OF BELIEFS IN ONLINE DEBATES

A Thesis

Submitted to the Faculty

of

Purdue University

by

Aldo Fabrizio Porco

In Partial Fulfillment of the

Requirements for the Degree

of

Master in Science

December 2019

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF THESIS APPROVAL

Dr. Dan Goldwasser, Chair
Department of Computer Science
Dr. Ming Yin
Department of Computer Science
Dr. Jean Honorio
Department of Computer Science

Approved by:
Dr. Clifton W. Bingham
Thesis Form Head

To all the people that supported me through this endeavour. Specially my father, my mother, my brother and my dear friend ML.

ACKNOWLEDGMENTS

Thanks to my advisor Dr. Dan Goldwasser for his teachings and support. Thanks to the Purdue NLP group for motivating me to learn about different areas and being a good environment for sharing and nurture ideas. Special thanks to Chang Li for collecting the *debates.org* dataset.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

ABBREVIATIONS

CMS	ChangeMyStance
CMV	ChangeMyView
NLP	Natural Language Processing
ILP	Integer Linear Programming
E2E	End-to End
HM	Hierarchical Modules
FF	Feed Forward neural networks

ABSTRACT

Aldo, Porco F. M.S., Purdue University, December 2019. Using Modular Architectures to Predict Change of Beliefs in Online Debates. Major Professor: Dan Goldwasser.

Researchers studying persuasion have mostly focused on modeling arguments to understand how people's beliefs can change. However, in order to convince an audience the speakers usually adapt their speech. This can be seen often in political campaigns when ideas are phrased - framed - in different ways according to the geographical region the candidate is in. This practice suggests that, in order to change people's beliefs, it is important to take into account their previous perspectives and topics of interest.

In this work we propose `ChangeMyStance`, a novel task to predict if a user would change their mind after being exposed to opposing views on a particular subject. This setting takes into account users' beliefs before a debate, thus modeling their preconceived notions about the topic. Moreover, we explore a new approach to solve the problem, where the task is decomposed into ”simpler” problems. Breaking the main objective into several tasks allows to build expert modules that combined produce better results. This strategy significantly outperforms a BERT end-to-end model over the same inputs.

1 INTRODUCTION

Convincing others is a skill highly cherished in our society. Salesman, politician and marketeer are just a few examples of professions that profit greatly on this trait. Persuasion has been used to denote the act of influencing people *beliefs*, motivations, attitudes and intentions [1]. Its aspects have been extensively studied [2, 3], giving shape to framing and rhetoric theory. The former is a schema of interpretation that relies on preconceived notions to communicate ideas [4], while the latter is defined as the art of persuasion through discourse [5]. Nowadays, the popularity of social platforms has allowed the act of persuasion to take place in public online spaces. Since this phenomenon happens on a massive scale, it has caught the eye of several researchers in this area [6–9].

The Natural Language Processing (NLP) community has studied beliefs changes under the presence of persuasive language. First, researchers have tried to identify beliefs under different settings like detecting legislation supporters in congressional speeches [10–12], or predicting stance in online debates [13–16]. These works motivated numerous studies that analyze the effectiveness of language when trying to change people's biases [17–20]. For example, Gleize et al. [17] suggest comparing the quality of two arguments based on a convincingness score. However, humans are biased, meaning that the efficacy of an argument is a function of its quality and their previous beliefs, morals and values. For example, a wealthy person might offer a lot of resistance to arguments in favor of increasing taxes to the rich.

Recently some research take into account people's beliefs when modeling persuasion. A good example is the study of the `/r/ChangeMyView` Reddit sub-community, where users describe their beliefs on a topic and invite others to change their minds. Tan et al. [18] created a classification task based on this mechanic and solve it by using clever argumentation and stylistic features. Having a detailed description of the users' belief over each subject seems contrived, but there are other ways of retrieving a similar signal. Nowadays there are many websites that track users' interactions with content and their peers. For example, Durmus et al. [21] create a dataset inspired by the `debate.org` website where they study convincingness by conditioning on users' political and religious ideology. To do so, they make use of self reported users' biases and differences in language content between debaters. However, adding more features without understanding the interaction of beliefs and arguments is not enough.

User behavior can attest to the characteristics that might make them susceptible to persuasion. Our approach, in opposition to similar studies [18, 21], models users' stance in several topics using self-reported traits and bias compatibility with their peers. The idea is that a better user representation will help identify biases and malleability. In order to prove our hypothesis we created a new task called *Change-MyStance* (CMS) that is based on the `debate.org` website's mechanics. It models the binary classification problem of deciding whether users will change their stance after joining a debate of two contenders with opposing views on a certain topic. One debater will agree with the voter's biases, while the other will not. The main contributions of the dataset are that it offers a more natural version of the problem of changing a person's beliefs in an online setting where (1) users have several opportunities to be persuaded on different topics, and (2) users can indirectly manifest their biases without explicitly stating their views. This setting allows to model the users in a wider beliefs spectrum.

Most works model the problem of changing people's stance as an end-to-end system [17, 18]. However, Our main hypothesis is that persuasiveness is a function of the arguments and the users' preconceived notions of the debated topic, thus we think modeling these two elements separately, i.e. we propose decomposing the task into sub-tasks that explicitly model beliefs and arguments. Particularly, we use the following sub-tasks:

- Agreement: shape user representation based on their shared beliefs.

- Before: predict users' bias prior to attending a debate.
- Arguments: given two pieces of text predict the one with better arguments.
- After: predict users' belief after being exposed to opposing views.

In order to solve the CMS task we establish the necessity to separate belief and argument detection. However, another important aspect is to understand their interaction. For example, a bias created by a trauma will be hard to change. There are several ways of combining the sub tasks that model each of this aspects. One way is to use a Multitask learning approach [22] where all sub-tasks are learned jointly and share some input representation. On the other hand, we propose a hierarchical structure based on modular learning, where each module corresponds to a task model. Naturally, there are sub-tasks that help alleviate a high level problem. The idea is to use domain knowledge as an heuristic to drive the hierarchical structure, i.e. each module should be directly supported by pertinent tasks. For example, a user belief After the debate should be a consequence of their beliefs Before reading the text and the quality of the Arguments the contenders make. Similarly, users change their minds when their beliefs Before and After being persuaded are different. We test two different strategies for building hierarchical structures: (1) connecting modules via a neural architecture and (2) defining constraints over their interaction using Integer Linear Programming (ILP). We empirically show that the neural hierarchical approach is more flexible and achieves better results than the alternatives.

We use the *ChangeMyStance* problem to test our hypothesis. First, we show the results of using different complexities of user representation and then we compare our modular approach against a strong end-to-end baseline using BERT [23], which is an attention based sentence encoder (sentences from the same paragraph are mapped close in the embedding space). Our results show the importance of modeling users' beliefs when predicting a change in stance and how using a modular approach can lead to statistically significant improvements.

2 RELATED WORK

How to convince people has been studied at least since 322 BC by the ancient Greeks. Aristotle defines the term rhetoric as *"the faculty of observing in any given case the available means of persuasion"* [24]. He built a framework for argumentative reasoning based on persuasive audience appeals (*logos, pathos and esos*) . After a few centuries, the Romans developed the theoretic principles of persuasive speech: invention, arrangement, style, memory and delivery. Nowadays, there are many books that describe persuasion with the focus of improving business and interpersonal relations [1, 25]. The term is usually defined as exerting influence on a person with the purpose of changing their views, attitudes, motivations and intentions [1]. We are particularly interested persuasive discourse, where it is important to create abstractions or metaphors that would allow a quicker understanding. This strategy is called framing [26] and it is used often in social movements as carriers of belief and ideology.

Persuasion has been a hot topic in the social sciences [2, 3], the Computational Linguistics [13, 14] and the Natural Language Processing [17, 18, 21] communities. Great efforts have been dedicated in understanding arguments to grasp the underlying mechanisms behind this phenomenon. They can be separated in argument characterization [19], automatic detection and component extraction [20, 27, 28], and quality assessment [6]. Gleize et al. [17] study it as way to improve the human-AI interaction by showing machines how to hold interesting discussions. They do so by creating a ranking task that given a pair of arguments decides which is more persuasive based on a convincingness score. However, we claim that understanding people's beliefs is key for persuasion to be successful, thus analyzing it only from the language perspective is not enough.

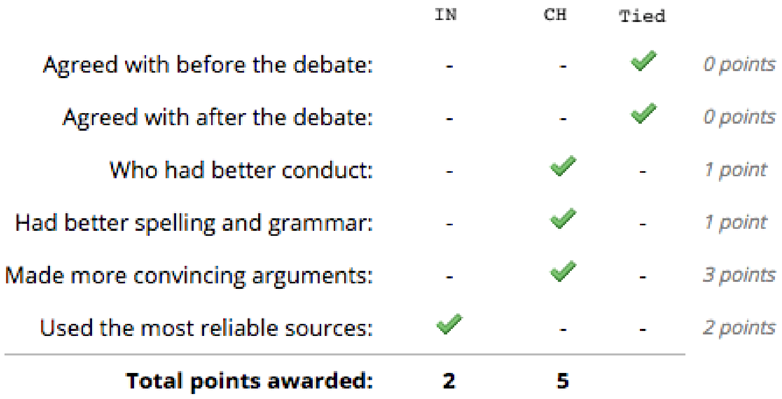
In order to change someone's belief to a target stance, it is necessary that the people being persuaded does not identify with it. The problem of predicting someone's bias over a certain topic is called the stance classification task. In its inception, researchers try to detect support (or opposition) towards a certain legislation in congressional speeches [10–12]. With the popularity of online social environments, the attention shifted towards predicting online debaters' biases in a given session (topic debate) based on how they argue [13–16]. Detecting people's beliefs and their strength is a first step in understanding if it is possible to convince them.

One of the ways persuasion has been studied is as the ability to change someone's stance. This problem was embodied by the `ChangeMyView` (CMV) task created by Tan et al. [18]. They built a dataset by scrapping the `/r/ChangeMyView` Reddit sub community where a member writes their belief about a topic and invites others to change their stance. Their setting includes a detailed written self-description of the thread creators' biases and posts from other users trying to convince them. The proposed solution consists on creating features corresponding to three categories: arguments being discussed by both stances, reasons being expressed only by one end, and stylistic features to measure malleability. Using the same dataset, Hidey et al. [29] showed the importance of taking into account the argumentation order, while Xiao et al [30] explored the effect of modeling user's psychological attributes.

The CMV dataset is an interesting way to study persuasion but its setup seems unrealistic as it is challenging to find people's detailed biases description for every topic in a more general setting. Moreover, people often overlook frames that are relevant to the strength of their belief, making statements that are not fateful to their biases. Similar to our work, Durmus et al [21] focus on modeling users' beliefs to predict a change in stance by conditioning on their religious convictions and political ideologies. They collected a dataset from `debate.org` extracting the 48 *big issues* (abortion legalization, minimum wage, etc) together with some profile information (age, country, education, etc) for each user. We pursue this idea further and model user's beliefs by taking into account a broader spectrum of user attributes, their association with arguments, and their belief agreement.

Even when the NLP community has been mostly using end-to-end models to predict changes in stance [17, 18], researchers found the need of decomposing the problem in simpler sub-tasks. In order to solve the CMV problem, Jo et al. [31] proposed a neural architecture with a modular attention mechanism that consists on two components: a model to detect user malleability and a module that identifies the relationship between opposing arguments. The idea is that the identified sub-tasks: (1) should be easier to solve than the high level problem, and (2) given that intuitively they are correlated with CMV, using them as input should help alleviate the burden of the main task. One of the main issues when working with a modular setting is to build an architecture that directs each signal where it is most useful. Research community has invested some efforts in creating a general modular framework [32] that automatically decides signal routing. However, we prove that domain knowledge can be effectively used as an heuristic to model module interaction and obtain results that improve our baselines.

3 THE DATASET



	IN	CH	Tied	
Agreed with before the debate:	-	-	✓	*0 points*
Agreed with after the debate:	-	-	✓	*0 points*
Who had better conduct:	-	✓	-	*1 point*
Had better spelling and grammar:	-	✓	-	*1 point*
Made more convincing arguments:	-	✓	-	*3 points*
Used the most reliable sources:	✓	-	-	*2 points*
Total points awarded:	**2**	**5**		

Figure 3.1. Example of the `debate.org` voting system. Each voter fills the same form. The rows correspond to different aspects/categories of the debate and the columns correspond to the available targets.

Our dataset was extracted from the `www.debate.org` website. Their mechanics allows users to express their change of view after a debate, and share prior beliefs in different ways. Each debate comprises on two contenders, one called the initiator (IN) and the other called the challenger (CH). The former creates a debate title, chooses the stance they want to promote, and writes a post arguing in its favor. The latter agrees to the challenge and contends the initiator's point of view. After their main arguments are laid out, both users have the necessary rounds to reason about each other's arguments. Until the voting period closes any user can attend the debate, read both point of views, and cast their votes in different aspects of the debate. Votes will give points to each contender depending on their target: 0 for IN, 2 for CH and 1 for tie.

Table 3.1.
Number of samples (votes) per class for each task.

Task	Class 0	Class 1	Class 2
Arguments	21259	7981	37442
Before	13947	33655	19080
After	13866	35989	16827
ChangeMyStance	93633	5851	-

An example of the voting system can be seen in Fig. 3.1, where columns correspond to the targets (IN, CH or tie) and rows to different aspects of the debate. The meaning of the categories are as follows:

- BEFORE: user's belief **before** the debate.
- AFTER: user's belief **after** the debate.
- ARGUMENTS: debater with better **arguments**.
- CONDUCT: debater with better **conduct**.
- WRITING: debater with better **writing** skills.

Another important factor of building a dataset based on `www.debate.org` is that it allows users to explicitly identify themselves based on beliefs and demographics information. The user profile $U_{profile}$ includes demographic information (like age, education and occupation) and their stance over the most controversial topics in the portal called the "big"-issues (see Table. 3.2). Finally, the summary section $U_{summary}$ is an open document that can be used by voters to further describe themselves.

3.1 Preprocessing

Our corpus consists of 12,901 debates and 51,096 users after removing noisy samples. In order to create the dataset, we only consider debates that contain text from

Table 3.2.
Big-issues defined in the users' profile.

Big-issues		
Abortion	Affirmative Action	Animal Rights
Barack Obama	Border Fence	Capitalism
Civil Unions	Death Penalty	Drug Legalization
Electoral College	Environmental Protection	Estate Tax
European Union	Euthanasia	Federal Reserve
Flat Tax	Free Trade	Gay Marriage
Global Warming Exists	Globalization	Gold Standard
Gun Rights	Homeschooling	Internet Censorship
Iran-Iraq War	Labor Union	Legalized Prostitution
Medicaid and Medicare	Medical Marijuana	Military Intervention
Minimum Wage	National Health Care	National Retail Sales Tax
Occupy Movement	Progressive Tax	Racial Profiling
Redistribution	Smoking Ban	Social Programs
Social Security	Socialism	Stimulus Spending
Term Limits	Torture	United Nations
War in Afghanistan	War on Terror	Welfare

both stances and at least three votes. Moreover, the text was further simplified by replacing URLS for a token ”$< url >$” and substituting strings that appeared together more than two times.

In order to represent each debater's text, we fine-tuned the `bert-base-uncased` model. BERT's objective is to give a similar representation to sentences in the same paragraph. In our case, we use the first 510 tokens of each round as a sentence and the concatenation of all rounds as paragraph. Since in a debate rounds belonging to the same writer will be close in the embedding space, we only use the first round encoding as the representation for the whole document (we use the same model to encode $U_{summary}$). Finally, we use a one-hot representation for the big-issues and most profile attributes that are discrete. However, we use $< lat, long >$ coordinates to represent the states the user lives in, and a continuous number to for their age.

4 THE TASK

ChangeMyStance is binary classification problem where the objective is to predict if users changed their minds after attending to a debate. An instance is defined as a 4-tuple $< u, T_{con}, T_{other}, y >$. We say that the user u is persuaded when their beliefs $before(u) \in \{0, 1, 2\}$ and $after(u) \in \{0, 1, 2\}$ the debate are different.

$$y = \begin{cases} 1, & \text{if } before(u) \neq after(u), \\ 0, & \text{otherwise} \end{cases}$$

The task is defined over a user u and two text representations T_{con} and T_{other}. In our setting both documents have opposite $stance(t) \in \{0, 2\}$

$$stance(T_{con}) \neq stance(T_{other})$$

Moreover, It is always true that U biases are different than T_{con} stance.

$$before(u) \neq stance(T_{con})$$

However, it does not imply that U beliefs agree with T_{other}'s views. For example, when the users' biases are tied ($before(u) = 1$), either text can change their stance.

4.1 Decomposing the ChangeMyStance Task

Changing someone's beliefs has been often solved as an end-to-end problem, but in this work we take a modular approach. Breaking down ChangeMyStance into subtasks gives several advantages. First, solving a single sub-task should be more simple than the high level problem. As a consequence, it should be natural to focus on the part of the input matter, thus making it easier to define the right setting and model for them. The result is having specialized models that contribute specific aspects of the CMS problem and alleviating its difficulty.

We have identified four different modules that help improve the results. The data statistics of each task can be found in Table 3.1.

4.1.1 Sub-task: Arguments

This task is inspired on the votes for the Arguments category of the voting scheme showed in Fig. 3.1. There users shows their preference y for the arguments written for the initiator $y = 0$, the challenger $y = 2$ or neither $y = 1$. Therefore, we define Arguments as a multi-class classification problem. An instance of the task is defined as a 4-tuple $< u, T_{in}, T_{ch}, y >$, where u is the voting user, T_{in} is the initiator's text, T_{ch} the challenger's text and $y \in \{0, 1, 2\}$ the user preference. It is important to notice that this problem is different than CMS. Even when a voter acknowledges that a debater has better arguments (than the one supporting their belief) it does not mean they where able to convince them. A perfect example of this is shown in Fig. 3.1 where the user was still undecided about their stance in spite of finding the challengers case more appealing.

4.1.2 Sub-task: Before

The Before task arises as a way to model users' beliefs using text as a proxy. It inspired on the self-reported preference over the stances of a debate *before* reading its content. The first row of the `debate.org` voting scheme (Fig. 3.1) corresponds to the problem's supervision. It is important to notice that this supervision was also used to define the CMS task and corresponds to the $before(u) \in \{0, 1, 2\}$ function. Moreover, the CMS problem already encodes the non-preferred stance in the order the inputs take in the instance, however this task allow the model to tune the user representation to capture this initial bias.

We will define Before as a multi-class classification problem where users' beliefs y were aligned with the initiator's T_{in}, the challenger's T_{ch} or no one's text before the debate. An instance of the task is a 4-tuple $< u, T_{in}, T_{ch}, y >$, where $y \in \{0, 1, 2\}$.

4.1.3 Sub-task: After

This task corresponds to the users' stance *after* taking into account the arguments written in a debate. Its supervision comes from the second row of the voting scheme shown in Fig. 3.1 and its used to define CMS in the form of a function $after(u) \in \{0, 1, 2\}$. Similar to the Before task, it is a multi-class classification problem where an instance is defined as a 4-tuple $< u, T_{in}, T_{ch}, y >$ and $y \in \{0, 1, 2\}$.

4.1.4 Sub-task: Agreement

Another way to model users' biases is creating an embedding that increases similarity between voters when they agree on their beliefs. This idea comes as a way to improve the representation of occasional users that have a very small vote count. The Agreement task uses Before's supervision $before(u) \in \{0, 1, 2\}$. We say that a user u_1 is similar to a user u_2 when they attended the same debate and $before(u_1) = before(u_2)$. An instance of this problem is a triplet of the form $< u_1, u_2, y >$.

$$y = \begin{cases} 1, & \text{if } before(u_1) = before(u_2), \\ 0, & \text{otherwise} \end{cases}$$

5 MODELS

So far we have argued the advantages of decomposing the ChangeMyStance problem and we have identified several subtasks that can help supporting it. However, one of our technical challenges is looking for ways to integrate signals from different models into a unified system that helps alleviate the CMS task. The approaches we have considered in this study are *Multitask Learning* [22], *Inference-based* and a *Modular Architecture*. Out of all the options we consider, the modular architecture approach is the most flexible one. It allows to explicitly define modules interaction using domain knowledge without the necessity of declaring complicated constraints between them. Moreover, this strategy should make the decision easy to interpret by understanding which signals are propagated from each module.

In the rest of this section we explain the different ways of composing sub-tasks. We explore the basic end-to-end model followed by our hierarchical approach for combining modules. We will conclude with a description of the multitask and inference-based approaches. But first we will briefly define define formally the concept of module compositionality.

In calculus [33], the composition operation $\circ$ is defined over two functions $m : X \to Y$ and $f : Z \to X$. Such operation is written as $m \circ f$ and its result is a new function h where $h(x) = m(f(x))$ and $h : Z \to Y$. Models are functions because they consistently map the inputs to the same outputs but their composisionality works slightly different. The main differences are:

- A sub module can transform only part of the parent module's input.
- The parent module is fed by the original input together and its sub-module's outputs.

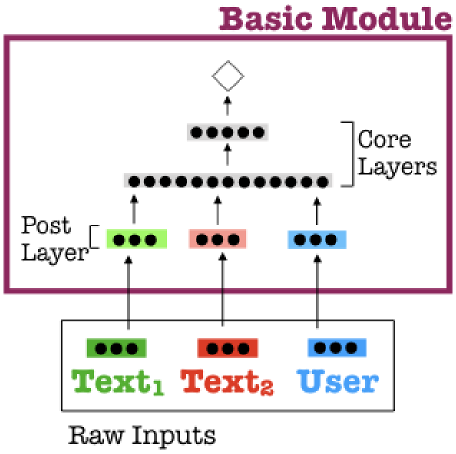


Figure 5.1. Graphical representation of the *ChangeMyStance* non-modular architecture. Similar architectures were used for Arguments, Before and Arguments.

We will write the composition of the parent module $P : X \rightarrow Y$ over the children sub-modules $[m_0, \ldots, m_n]$, where $m_i : X \rightarrow Z_i$, as:

$$P[m_0, \ldots, m_n] = H(X, m_0(X), \ldots, m_n(X))$$

H is a new module where $H : X, Z_0, \ldots, Z_n \rightarrow Y$.

As an example, we can say that the users' preference after a debate depends on their previous beliefs and the power of the debaters' arguments. This idea can be write as a compositional module AFTER that is supported by the BEFORE and ARGUMENTS sub modules:

$$\textsc{After}[\textsc{Before}, \textsc{Arguments}]$$

Now, lets use the CMS definition. Voters changed their minds if their views BEFORE a debate were different than their views AFTER. Applying the compositionality definition over this idea we can create the module in Eq. 5.1.

$$\text{CMS}[\textsc{After}[\textsc{Before}, \textsc{Arguments}], \textsc{Before}] \tag{5.1}$$

5.1 End-to End model

The End-to End (E2E) model is a feed forward (FF) neural network. As its names suggests, it performs a direct mapping from input to outputs without using other modules. The neural architecture, as shown in Fig. 5.1, is divided in two parts:

- *Post Layer*: is composed of a single hidden layer and its objective is to reduce the text embedding dimensionality. It is applied twice per instance because the input of our tasks is comprised by two BERT embeddings (one for each stance).
- *Core Layers*: consist on two FF hidden layers. They are fed with the concatenation of the user embedding and the stance text's hidden representation (output of the *post layer*).

Since ChangeMyStance, Arguments, Before and After instances comprise the same input, the E2E model suits them. Moreover, all the aforementioned tasks are classification problems, thus the output of the *Core Layers* is used as input to a *Softmax* layer that provides the final decision. As we will explain in the next section, end-to-end models can be used by other modules.

5.2 Hierarchical Models

The easiest way to understand Hierarchical Models (HM) is to think about them as end-to-end models that also receive information from other sources - in our case, modules. HM have two types of inputs depending if they come from the original task or from a supporting module. As can be seen in in Fig. 5.2, if there is no information coming from sub-modules, the HM takes the same form as its E2E version. On the other hand, if it receives input from another module its architecture will consist of three parts:

- *Post Layer*: has the same architecture and works in the same way as the E2E model.

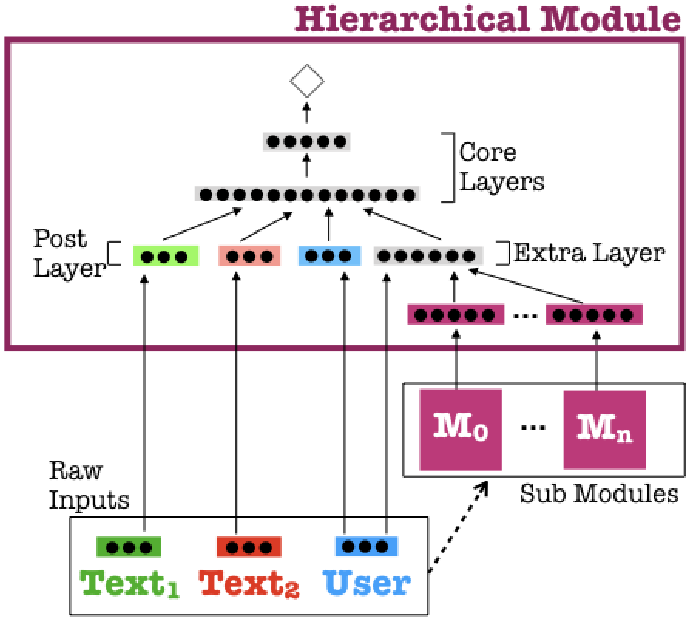


Figure 5.2. Graphical representation of a Hierarchical Model.

- *Extra Layer*: is comprised by one FF layer. It receives as input the concatenation of the user encoding together with the last hidden layer representation of each supporting module.

- *Core Layers*: has the same functionality and structure as the E2E model, but with the subtle difference that it also receives as input the *Extra Layer* hidden representation.

There are many ways in which HM can use their sub-modules signal. In this paper we pre-train each sub-module and use their last hidden layer encoding as fixed inputs to the downstream module. Intuitively, the last layer represent the final input transformation, containing the information needed for the sub-module classification decision. Another interesting way would be to attach their neural architecture and back-propagate the error from the HM back to the sub-modules. In practice we found this method to be very slow and we will leave this idea open as further work.

It is important to notice that the user encoding is used separately by the core layers and the extra layer. The hypothesis behind this architectural choice is that

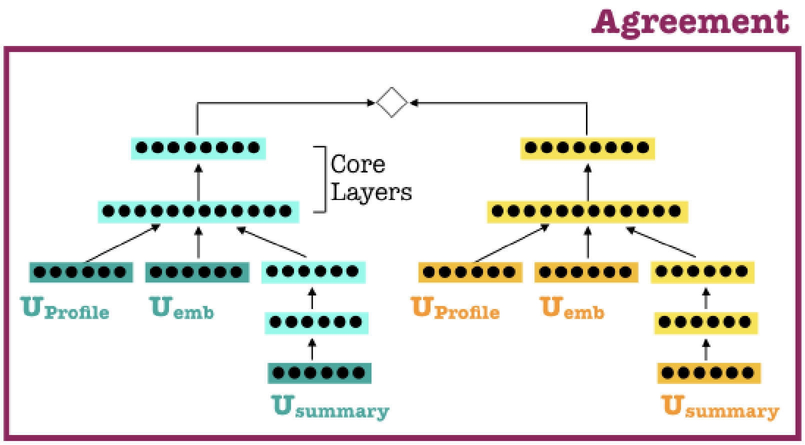


Figure 5.3. Graphical representation of the Agreement module.

the sub-modules' weight should be dependent on the voter. For example, a "used trusted references" module would probably be very important to a voter with a high education level.

We will write hierarchical modules using the composition operator defined at the start of this chapter. A HM $P[m_0, \ldots, m_n]$ is defined as $H(X, m_0(X), \ldots, m_n(X))$, where the instance X will be processed by the E2E part, and the rest of the inputs $m_0(X), \ldots, m_n(X)$ will be handled by the *Extra layer*.

5.3 Agreement model

The Agreement module emerges as a way to fight vote sparsity in the data. The idea is to create an embedding space where users are closer together when they have similar beliefs. This will help create better representations for casual voters that do not have enough participation. As shown Fig. 5.3, the Agreement task receives as input two user representations comprised by $U_{profile}$, U_{sumary} and U_{emb} (randomly initialized user embedding). The components of its architecture are devided in:

- The *Summary Encoding Layers* are two FF hidden layers that take as input the BERT embedding of the user summary and outputs a lower dimensionality representation of it.

- The *Core layers Layers* are two FF hidden layers that take as input the concatenation of the *Summary Encoding Layers* hidden representation together with U_{emb} and $U_{profile}$, and outputs a user embedding.

The AGREEMENT neural architecture does not feed a final *Softmax* layer. Instead, because we have an embedding objective, the network is separately applied over the two user representations and compared using the euclidean distance. We use the *Hinge Embbeding Loss* as training objective - defined in Eq. 5.2 - to quantify the similarity errors. In the loss definition, $y = 1$ means that users are similar (expressed the same belief in a debate) while $y = 0$ means they are dissimilar. It is important to notice that users without a polarized belief (”tie” votes) where not include in training. This was done in order to avoid introducing too much noise in the embedding space.

$$l_n = \begin{cases} x_n, & \text{if } y_n = 1, \\ \max\{0, \Delta - x_n\}, & \text{if } y_n = 0 \end{cases} \tag{5.2}$$

5.4 Multitask model

As said before, our modular approach pre-trains the modules and then uses their last hidden state as instance representation for the task. However, we also wanted to experiment how would a joint model behave in the same setting. For that reason, we built a multitask model where all tasks are trained at the same time and share parts of the architecture. As can be seen in Fig. 5.4, the *Multitask model* comprises the following elements:

- The *Post Layer*, which works in the same way as the E2E model. This layer together with the user representation are shared between all tasks.

- The *Core Layers*: work in the same way as the end-to-end model and each task has his own.

By using this paradigm, each task contributes to shaping the inputs representation, possibly helping other tasks in the process. Moreover, we use the sum of the Cross Entropy Loss for each task as classification objective. Specifically, each objective has the same weights as the others, which should not let tasks overpower each other.

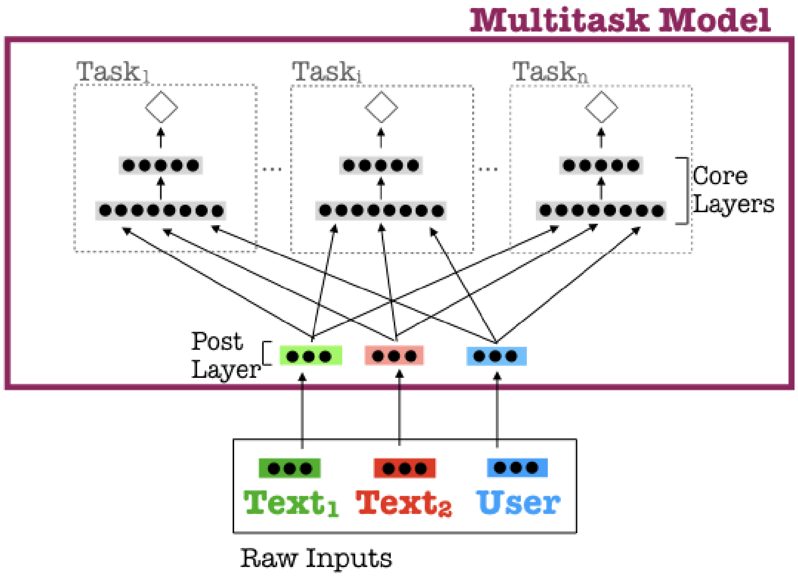


Figure 5.4. Graphical representation of the multitask model.

5.5 ILP based Inference

So far we have used neural architectures to capture the interaction between modules. However, enforcing consistency rules at test time between tasks is another way to achieve the same effect. Examples of interesting sub-tasks combinations that can act as a proxy for the CMS task are:

$$\textsc{Arguments} \neq \textsc{Before} \implies \texttt{ChangeMyStance} \tag{5.3}$$

$$\textsc{After} \neq \textsc{Before} \implies \texttt{ChangeMyStance} \tag{5.4}$$

$$\text{E2E CMS} \implies \texttt{ChangeMyStance} \tag{5.5}$$

Each defined rule represents an alternative definition of the ChangeMyStance task based on some sub-tasks. Rule 5.3 portrays the idea that if the users' arguments preference defers with their bias, then they changed their minds. On the other hand, the Rule 5.4 uses our definition of CMS, i.e. if the beliefs before the debate are not the same as the beliefs after the debate it means the users changed their stance. Finally, we incorporate the end-to-end task as a way to enforce consistence with the most simple version of the model.

There are several ways to model each rule. One of them would be to create constraints based on the single modules, but defining a big number of constraints would be necessary. An easier way is building hierarchical modules to predict ChangeMyStance based on the modules used in each rule:

$$\begin{aligned} \text{CMS}[\textsc{Arguments}, \textsc{Before}] &\implies \texttt{ChangeMyStance} \\ \text{CMS}[\textsc{Before}, \textsc{After}] &\implies \texttt{ChangeMyStance} \\ \text{CMS} &\implies \texttt{ChangeMyStance} \end{aligned}$$

For an specific instance, the weight associated with each rule is proportional to the probability assigned to the prediction of each hierarchical module.

We will need to enforce intra-rule and inter-rule constraints. The former restrict the values taken by variables created for a single rule, while the latter enforces consistency between rules. Given that CMS is a binary classification problem, each rule will have two variables, one for each class. The intra-rule constraints will enforce that only one class must be active for a given instance. In ILP terminology, the variables representing the classes for a single rule should sum one:

$$X_{rule,class_0} + X_{rule,class_1} = 1 \tag{5.6}$$

On the other hand, the inter-rule constraints should ensure that the active class is the same among all rules. In ILP terminology, the difference between variables representing the same class should be zero:

$$X_{rule_i,class_0} - X_{rule_j,class_0} = 0, \text{ where } rule_i \neq rule_j \tag{5.7}$$

Given that we want to maximize the probability of a instance being CMS, we propose as objective the sum of all variables weighed by their probability - given by the Hierarchical modules. Such objective is written in Eq. 5.8, where, i represents an instance, r a rule, c a class and $P(x)$ the c probability of i given by r.

$$max \sum_{r} \sum_{c} \sum_{i}^{N} P_{r,c,i} X_{r,c,i} \tag{5.8}$$

6 EXPERIMENTAL EVALUATION

In this chapter we will revisit our main research questions and show the empirical results supporting our findings. First we will enumerate the parameters used for each model and briefly describe how our experiments are run. Then we will explain the importance of the user in the ChangeMyStance task by incrementally analyzing richer representations. Finally, we will compare different compositional approaches measuring their performance against end to end approaches.

6.1 Models Setup

Our general experimental setting consists in running a 10-fold cross validation on the debates, meaning that votes from the same debate will end in the same fold. We use one fold for testing. From the remaining 9-folds, we randomly choose 15% of debates for validation (20% for Arguments) and the rest for training. We use Adam [34] as iterative method to optimize the *Cross Entropy Loss* and the *Hinge Embedding Loss* for classification and embedding tasks respectively. Given that one class usually is significantly grater than the other (see Fig. 3.1) we balance their weight in the objective functions. We fix the learning rate to a value of 0.0001 (0.001 for Agreement) and training stops if there is no improvement after 25 (50 for Agreement) epochs for a maximum of 200 epochs. An epoch consists on a full pass of the training samples arranged into batches of size 256.

6.1.1 Raw Inputs

The BERT encoding has a size of 9216 features and it is used for each stance text. The user representation (unless explicitly stated) is built from the concatenation of its profile $U_{profile}$ and a randomly generated user embedding U_{emb} of size 158 and 100 (50 for *Arguments*) respectively. The user summary $U_{summary}$ was used only as part of the Agreement module and it is encoded by the BERT model.

6.1.2 Architectures

In all our experiments we employ Neural Networks to build modules. All the layers the models contain are feedforward (their connections do not form cycles) and the activation functions are Sigmoid. Moreover, the described models are built using a subset of the following parts: a *Post Layer* of size 100, two *Core Layers*, each of size 50, and an *Extra Layer* with size equal to the size of its inputs.

We have three special architectures that require further specification:

- The Agreement module works as an embedding model that maps a user to a belief space. It comprises two *Summary Encoding Layers* of size 50 and two *Core Layers* of size 100.

- The Multitask module uses the sum of each task's Cross Entropy loss as the objective function. Moreover, samples from all task are shuffled and they are put together in batches of size 256 for training.

- The Inference-based model (ILP) selects the best weights for each rule based on the validation set performance. In all of our folds the best set of weights picked at validation time were 0.9, 0.1 and 0.1 for rules 5.3, 5.4 and 5.5 respectively shown in the Models chapter.

Table 6.1.
Avg. F1-score and +F1 (F1-score of the positive class) for the E2E model using different user representations. All numbers are truncated to the 3rd decimal. We test statistical significance "*" with $p - value < 0.01$ over the closest simplified version. For Example: "Text + $U_{profile}$ + U_{emb}" is tested against "Text + $U_{profile}$".

Task	Text		Text + $U_{profile}$		Text + $U_{profile}$ + U_{emb}	
	Validation	Test	Validation	Test	Validation	Test
	Avg. F1	Avg. F1	Avg. F1	Avg. F1	Avg. F1	Avg. F1
Arguments	0.5164	0.4913	0.522	0.494	0.534	0.507*
Before	0.4364	0.4264	0.473	0.464*	0.531	0.517*
After	0.4494	0.4389	0.486	0.475*	0.548	0.534*
CMS	0.499	0.498	0.533	0.530*	0.606	0.601*
	+F1	+F1	+F1	+F1	+F1	+F1
CMS	0.174	0.170	0.203	0.197*	0.282	0.272*

6.2 Experimental Design and Results

6.2.1 The importance of user biases

One of our motivations is to show how modeling users' biases is key in predicting beliefs changes. In order to prove this idea we trained the ChangeMyStance task and some of its supporting sub-tasks with increasingly richer user representations:

- *Text*: only using the BERT embeddings, i.e. no user representation.
- *Text*+$U_{profile}$: *Text* with demographics and big-issues features.
- *Text*+$U_{profile}$+U_{emb}: *Text*+$U_{profile}$ with a randomly initialized user embedding.

As can be seen in Table 6.1, a richer user representation yields significant improvements. This is specially true in tasks modeling users beliefs as ChangeMyStance, Before and After in contrast with the Arguments task. The latter tasks in-

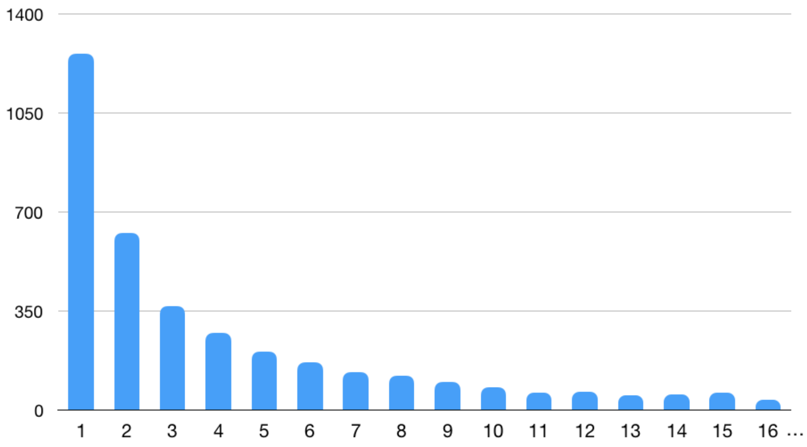


Figure 6.1. Histogram of the number of votes per user.

crease approximately 3 points when adding the user profile, and around 6 points when adding the randomly generated user embedding. On the other hand, the Arguments E2E model barely benefits from the richer representations, which means that this task is mostly dependent on the debates' content.

One could think that because most users are casual (Fig. 6.1), creating a randomly initialized user embedding would not make a considerable change. As we can see in the last column of Table. 6.1, that is not the case. Adding U_{emb} produces a substantial performance increase in all the end to end models. Our hypothesis is that it help characterize the users that have several votes.

Now that we have shown that users' beliefs play an important role in the CMS problem, it would be interesting to include a module that models biases from a user perspective. The Agreement task is the embodiment of this idea. In Table 5.2 we can evaluate the relevance of using Agreement E2E module for supporting the Before task. As can be seen, the hierarchical module Before[Agreement] is significantly better than its end-to-end version Before. Our hypothesis is that users with a small number of votes are harder to characterize. However, they will most probably attend to debates where frequent users participate. The more they vote together, the better their representation should become.

Table 6.2.
Predicting change of stance based on ideology agreement of Conservatives and Liberals. The study uses as metrics the percentage of the positive class (%+), the F1-score of the positive class (+F1) and the Avg. F1-score.

Ideology		Metric		
Voter	Writer	%+	+F1	Avg. F1
Liberal	Liberal	7.8	0.279	0.600
Liberal	Conserv.	3.7	0.289	0.602
Conserv.	Liberal	6.6	0.293	0.607
Conserv.	Conserv.	15.9	0.405	0.647

We provide additional analysis based on users ideology in Table 6.2. We compare our model's performance based on the similarity between the ideology of the voting user and the contender, aiming to change the voter's mind. Interestingly, the task is significantly easier when both users have a conservative ideology.

6.2.2 Compositionality and Modular Learning

In this section we will show that the idea of decomposing a high level task can make a significant difference in the models' final performance. Also, we will make a case for hierarchical modules by comparing them to other compositional approaches. Finally, we will test if using domain knowledge as an heuristic to build modules interaction produces any improvements.

Task decomposition

In order to prove that finding relevant sub-tasks to alleviate the downstream task is useful, we measure the performance of the simplest hierarchical modules - only two levels - against their end-to-end versions. We can corroborate in Table 6.4 that

Table 6.3.
Avg. F1 scores of the end-to-end (E2E) and Hierarchical models for the supporting tasks. If $p - value < 0.01$ then * when comparing basic and hierarchical.

Models	Validation	Test
	Avg. F1	Avg. F1
E2E Agreement	0.528	0.520
E2E Arguments	0.534	0.507
E2E Before	0.531	0.517
Before[Agreement]	0.544	0.533*
E2E After	0.548	0.534
After[Before[Agreement], Arguments]	0.562	0.550*

the ChangeMyStance task gets a statistically significant improvement when all sub-tasks are supporting it - CMS[After, Before, Arguments, Agreement] model. Moreover, we can confirm that decomposition also helps sub-task. In Table 6.3 we can see how the hierarchical model Before[Agreement] produces an statistical significant increase over its E2E version.

Domain Knowledge

We previously showed how researchers have been able to improve results by enhancing signal routing [32]. The intuition is that if the relevant information travels without interference to the point where it is useful, then it will make the biggest difference. In our case, we have signal coming from each module and it would take exponential time to test every possible composition. Therefore, we are looking heuristics to build an architecture that would boost performance. That is where domain knowledge comes into play. We compiled a list of reasons why one task should support another and tested our hypothesis:

- Before should use Agreement as sub-module because understanding which users have similar beliefs is critical in predicting biases. As shown in Table. 6.3, Before[Agreement] is significantly better that E2E Before.

- The beliefs After the debate are a consequence of how strong the bias in the voter is, and the *Arguments* used to persuade them. Looking at Table. 6.3 we can corroborate that the end-to-end After model is outperformed by its hierarchical version using Before[Agreement] and Arguments as auxiliary sub-modules After[Before[Agreement], Arguments].

- As said before, the ChangeMyStance task is defined as the users' beleifs being different Before and After a debate. Therefore, Before and After should always support CMS. As can be seen in Table 6.4 all the CMS hierarchical models that are supported by these tasks achieve better results than the E2E CMS model.

Comparing compositional approaches

Now we want to test different strategies to combine modules. First we will compare our modular approach embodied by the hierarchical model with the multitask setting. The former separately pre-trains each module and they do not share parameters. On the other side, the multitask approach jointly trains all task while they share part of the architecture. We compare their performance in solving the ChangeMyStance task while being supported by the Before, After and Arguments sub-tasks. As can be seen in Table 6.4, the hierarchical model CMS[Before, After, Arguments] performs significantly better than its E2E version (and its contender), while the Multitask - [CMS, Before, After, Arguments] does not. We think the reason why the multitask approach performs poorly is that objectives tend to fight over the shared representation, making learning harder. Problem that is avoided with the modular representation.

It is also possible to combine modules at inference time using ILP. In order to do this we defined some rules using domain knowledge as an heuristic. Moreover, we used hierarchical modules to materialize this rules and created constraints to enforce their agreement.

- The first rule 5.3 was embodied by the hierarchical model CMS[BEFORE[AGREE], AFTER[BEFORE[AGREE], ARGS]] which uses BEFORE and AFTER as the CMS supporting modules.
- The second rule 5.4 tries to predict CMS as a consequence of the ARGUMENTS preference being different than the beliefs. The idea is materialized by the module CMS[BEFORE[AGREE], ARGS].
- The third rule 5.5 is built by using the E2E CMS module.

In order to compare the ILP and the modular approaches on equal ground, we used the modules defined by each rule as auxiliary tasks of the CMS hierarchical module, i.e. we are using the neural architecture to perform inference. As can be seen in Table. 6.4, the ILP model manages to outperform the CMS end to end model, feat that was not achieved by the multitask learning approach. However, the modular method seems to be better than the ILP when predicting the positive CMS class.

Modular Composition

Now, we want to compare different modular architectures to understand the impact of their structure in the performance. In order to test this idea we compare two CMS hierarchical modules:

- A depth two HM that uses CHANGEMYVIEW as the high level task and all end-to-end version of the supporting sub tasks CMS[AFTER, BEFORE, ARGS, AGREE]
- A hierarchical module that uses the best version of all supporting tasks, i.e. instead of using their E2E version we will use their best hierarchical version. More

Table 6.4.
Results of using different strategies when shaping the modules interaction to predict CMS. We use the Average F1 score and the *+F1* (the F1-score of the positive CMS class) metrics to characterize the models. We test for statistical significance with $p - value < 0.05$: (1) ”*” w.r.t E2E CMS, (2) ”$^\cdot$” w.r.t Multitask, (3) ”$^+$” w.r.t ILP, and (4) ”$^-$” w.r.t CMS[After, Before, Args, Agree].

Models	Validation		Test	
Baselines	+F1	Avg. F1	+F1	Avg. F1
E2E CMS	0.282	0.606	0.272	0.601
Multitask[CMS, Before, After, Args]	0.283	0.602	0.278	0.599
ILP[CMS, CMS[Before[Agree], After[Before[Agree], Args]], CMS[Before[Agree], Args]]	0.283	0.605	0.279^*	0.603^*
Hierarchical Models	+F1	Avg. F1	+F1	Avg. F1
CMS[Before, After, Args]	0.297	0.609	$0.288^\cdot$	0.605
CMS[After, Before, Args, Agree]	0.296	0.607	0.286^*	0.602
CMS[CMS[Before[Agree], After[Before[Agree], Args]], CMS[Before[Agree], Args]]	0.293	0.608	0.284^*	0.603
CMS[After[Before[Agree], Args], Before[Agree], Args, Agree]	0.299	0.612	0.292^{+-}	0.608^{+-}

explicitly, the auxiliary sub-modules will be After[Before[Agreement], Arguments, Before[Agreement], E2E Agreement and E2E Arguments. The final model will be written as

CMS[After[Before[Agree], Args], Before[Agree], Args, Agree].

If the modules do not benefit from each other both modules should obtain similar results. However, as we can see in Table 6.4, the hierarchical module using the hierarchical sub-modules is significantly better than using their end to end versions model. In other words, building hierarchies based on improved sub-modules can improve the overall performance.

7 SUMMARY

This work presents another perspective of the problem of changing peoples' stance in online debates. We built version of the task where people declare their beliefs in different topics, through different mechanics and in a more natural way. Our main insight lies in the fact that the convincingness of arguments depends on the of the listener's preconceived notions about the topic, i.e. a person with a strong bias will be harder to persuade. This intuition leads us to identify tasks - strongly related with the problem of changing someone's mind - that represent users' beliefs and discourse arguments, and study their interaction.

In order to prove our hypothesis we build a new dataset ChangeMyStance based on *debate.org* mechanics. There, users can manifest their stances in direct or indirect ways, and their arguments preferences through the voting scheme when attending a debate. We identify four tasks that are highly related with the CMS problem, and used them to alleviate its difficulty: Before, After, Arguments and Agreement. We propose different compositional approaches to represent the tasks' interactions and compare with an end-to-end approach. Modular learning - our approach - is embodied by a hierarchical model. We compare it with two solutions often found in the literature: Multitask Learning and ILP based inference. Their main difference is that the modular strategy conditions in the latent representation of the modules' output, the multitask approach conditions on the input representation and the ILP setting conditions on the outputs. Particularly, the ILP uses hard constraints to model the tasks' interaction while the hierarchical model uses a neural architecture.

We trained E2E models using increasingly complex user representations. From this experiments we learned that increasing the user complexity produces a better performance of the tasks that (intuitively) should highly depend on modeling user's beliefs (CMS, Before and After). This hunch was corroborated when the Agreement module improved the performance of the Before task. However, we empirically verified that a better user representation does not make a big difference when predicting the Arguments task, probably because it depends on text.

By running experiments with and without supporting modules we could confirm that the identified sub-tasks are relevant for the CMS problem. Moreover, we tested our compositional strategies showing that the Modular Learning approach is better in comparison to the Multitask and ILP based approaches when run using the same supporting modules. Finally, we used domain knowledge to reason about the hierarchical structure of the modules, and we demonstrated that it can be effectively used as an heuristic to improve signal routing in a modular architecture.

REFERENCES

[1] Robert H. Gass and John S. Seiter. *Persuasion : social influence and compliance gaining*. Pearson, 2014.

[2] B. J. Fogg and B.J. *Persuasive technology : using computers to change what we think and do*. Morgan Kaufmann Publishers, 2003.

[3] Samuel L. Popkin. *The reasoning voter : communication and persuasion in presidential campaigns*. University of Chicago Press, 1991.

[4] Erving Goffman. *Frame analysis : an essay on the organization of experience*.

[5] Edward P. J. Corbett and Robert J. Connors. *Classical rhetoric for the modern student*. Oxford University Press, 1999.

[6] Ivan Habernal and Iryna Gurevych. What makes a convincing argument? empirical analysis and detecting attributes of convincingness in web argumentation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1214–1223, Austin, Texas, November 2016. Association for Computational Linguistics.

[7] Peter Potash and Anna Rumshisky. Towards debate automation: a recurrent model for predicting debate winners. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2465–2475, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[8] Cristian Danescu-Niculescu-Mizil, Lillian Lee, Bo Pang, and Jon Kleinberg. Echoes of power: Language effects and power differences in social interaction. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 699–708, New York, NY, USA, 2012. ACM.

[9] Kazi Saidul Hasan and Vincent Ng. Why are you taking this stance? identifying and classifying reasons in ideological debates. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 751–762, Doha, Qatar, October 2014. Association for Computational Linguistics.

[10] Matt Thomas, Bo Pang, and Lillian Lee. Get out the vote: Determining support or opposition from congressional floor-debate transcripts. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 327–335, Sydney, Australia, July 2006. Association for Computational Linguistics.

[11] Mohit Bansal, Claire Cardie, and Lillian Lee. The power of negative thinking: Exploiting label disagreement in the min-cut classification framework. In *Proceedings of COLING: Companion volume: Posters*, pages 15–18, 2008.

[12] Clinton Burfoot, Steven Bird, and Timothy Baldwin. Collective classification of congressional floor-debate transcripts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1506–1515, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[13] Marilyn A. Walker, Pranav Anand, Robert Abbott, and Ricky Grant. Stance classification using dialogic properties of persuasion. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12, pages 592–596, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[14] Pranav Anand, Marilyn Walker, Rob Abbott, Jean E. Fox Tree, Robeson Bowmani, and Michael Minor. Cats rule and dogs drool!: Classifying stance in online debate. In *Proceedings of the 2Nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*, WASSA '11, pages 1–9, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[15] Swapna Somasundaran and Janyce Wiebe. Recognizing stances in online debates. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 226–234, Suntec, Singapore, August 2009. Association for Computational Linguistics.

[16] Amita Misra, Pranav Anand, Jean E. Fox Tree, and Marilyn Walker. Using summarization to discover argument facets in online idealogical dialog. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 430–440, Denver, Colorado, May–June 2015. Association for Computational Linguistics.

[17] Martin Gleize, Eyal Shnarch, Leshem Choshen, Lena Dankin, Guy Moshkowich, Ranit Aharonov, and Noam Slonim. Are You Convinced? Choosing the More Convincing Evidence with a Siamese Network, 2019.

[18] Chenhao Tan, Vlad Niculae, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. Winning Arguments. In *Proceedings of the 25th International Conference on World Wide Web - WWW '16*, pages 613–624, New York, New York, USA, 2016. ACM Press.

[19] Douglas Walton. Argument mining by applying argumentation schemes. *Studies in Logic*, 4, 04 2012.

[20] Raquel Mochales Palau and Marie-Francine Moens. Argumentation mining: The detection, classification and structure of arguments in text. In *Proceedings of the 12th International Conference on Artificial Intelligence and Law*, ICAIL '09, pages 98–107, New York, NY, USA, 2009. ACM.

[21] Esin Durmus and Claire Cardie. Exploring the role of prior beliefs for argument persuasion. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1035–1045, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[22] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural Language Processing (Almost) from Scratch. *J. Mach. Learn. Res.*, 12, nov 2011.

[23] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[24] ARISTOTLE. *RHETORIC.* SIMON & BROWN, 2018.

[25] Robert B. Cialdini. *Influence : the psychology of persuasion.*

[26] David Snow and Robert Benford. Ideology, frame resonance and participant mobilization. *International Social Movement Research*, 1:197–217, 01 1988.

[27] John Lawrence and Chris Reed. Combining argument mining techniques. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 127–136, Denver, CO, June 2015. Association for Computational Linguistics.

[28] Christian Stab and Iryna Gurevych. Parsing argumentation structures in persuasive essays. *Comput. Linguist.*, 43(3):619–659, September 2017.

[29] Christopher Hidey and Kathleen McKeown. Persuasive Influence Detection: The Role of Argument Sequencing. In *AAAI Conference on Artificial Intelligence*, 2018.

[30] Lu Xiao and Taraneh Khazaei. Changing Others' Beliefs Online. In *Proceedings of the 10th International Conference on Social Media and Society - SMSociety '19*, pages 92–101, New York, New York, USA, 2019. ACM Press.

[31] Yohan Jo, Shivani Poddar, Byungsoo Jeon, Qinlan Shen, Carolyn Rose, and Graham Neubig. Attentive Interaction Model: Modeling Changes in View in Argumentation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 103–116, Stroudsburg, PA, USA, 2018. Association for Computational Linguistics.

[32] Ignacio Cases, Clemens Rosenbaum, Matthew Riemer, Atticus Geiger, Tim Klinger, Alex Tamkin, Olivia Li, Sandhini Agarwal, Joshua D. Greene, Dan Jurafsky, Christopher Potts, and Lauri Karttunen. Recursive routing networks: Learning to compose modules for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3631–3648, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[33] Dale E. Varberg, Edwin J. (Edwin Joseph) Purcell, and Stephen. Rigdon. *Calculus : early transcendentals.* Pearson/Prentice Hall, 2007.

[34] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.