Cross-Domain Label-Adaptive Stance Detection

Hardalov, Momchil; Arora, Arnav; Nakov, Preslav; Augenstein, Isabelle

Published in:

DOI:
10.18653/v1/2021.emnlp-main.710

Publication date:
2021

Document version
Publisher's PDF, also known as Version of record

Document license:
Other

Citation for published version (APA):
Cross-Domain Label-Adaptive Stance Detection

Momchil Hardalov\textsuperscript{1,2} Arnav Arora\textsuperscript{1,3} Preslav Nakov\textsuperscript{1,4} Isabelle Augenstein\textsuperscript{1,3}

\textsuperscript{1}Checkstep Research
\textsuperscript{2}Sofia University “St. Kliment Ohridski”, Bulgaria
\textsuperscript{3}University of Copenhagen, Denmark
\textsuperscript{4}Qatar Computing Research Institute, HBKU, Doha, Qatar

\{momchil, arnav, preslav.nakov, isabelle\}@checkstep.com

Abstract

Stance detection concerns the classification of a writer’s viewpoint towards a target. There are different task variants, e.g., stance of a tweet vs. a full article, or stance with respect to a claim vs. an (implicit) topic. Moreover, task definitions vary, which includes the label inventory, the data collection, and the annotation protocol. All these aspects hinder cross-domain studies, as they require changes to standard domain adaptation approaches. In this paper, we perform an in-depth analysis of 16 stance detection datasets, and we explore the possibility for cross-domain learning from them. Moreover, we propose an end-to-end unsupervised framework for out-of-domain prediction of unseen, user-defined labels. In particular, we combine domain adaptation techniques such as mixture of experts and domain-adversarial training with label embeddings, and we demonstrate sizable performance gains over strong baselines, both (i) in-domain, i.e., for seen targets, and (ii) out-of-domain, i.e., for unseen targets. Finally, we perform an exhaustive analysis of the cross-domain results, and we highlight the important factors influencing the model performance.

1 Introduction

There are many different scenarios in which it is useful to study the attitude expressed in texts, e.g., of politicians with respect to newly proposed legislation (Somasundaran and Wiebe, 2010), of customers regarding new products (Somasundaran and Wiebe, 2009), or of the general public towards public health measures, e.g., aiming to reduce the spread of COVID-19 (Hossain et al., 2020; Glandt et al., 2021). This task, commonly referred to as stance detection, has been studied in many different forms: not just for different domains, but with more substantial differences in the settings, e.g., stance (i) expressed in tweets (Qazvinian et al., 2011; Mohammad et al., 2016; Conforti et al., 2020b) vs. long news articles (Pomerleau and Rao, 2017; Ferreira and Vlachos, 2016) vs. news outlets (Stefanov et al., 2020) vs. people (Darwish et al., 2020), (ii) with respect to a claim (Chen et al., 2019) vs. a topic, either explicit (Qazvinian et al., 2011; Walker et al., 2012) or implicit (Hasan and Ng, 2013; Gorrell et al., 2019). Moreover, there is substantial variation in (iii) the label inventory, in the exact label definition, in the data collection, in the annotation setup, in the domain, etc. The most crucial of these, which has not been investigated, currently preventing cross-domain studies, is that the label inventories differ between the settings, as shown in Table 1. Labels include not only variants of agree, disagree, and unrelated, but also difficult to cross-map ones, such as discuss and question.

Our goal in this paper is to design a common stance detection framework to facilitate future work on the problem is a cross-domain setting. To this end, we make the following contributions:

- We present the largest holistic study of stance detection to date, covering 16 datasets.
- We propose a novel framework (MoLE) that combines domain-adaptation and label embeddings for learning heterogeneous target labels.
- We further adapt the framework for out-of-domain predictions from a set of unseen targets, based on the label name similarity.
- Our proposed approach outperforms strong baselines both in-domain and out-of-domain.
- We perform an exhaustive analysis of cross-domain results, and find that the source domain, the vocabulary size, and the number of unique target labels are the most important factors for successful knowledge transfer.

Finally, we release our code, models, and data.\textsuperscript{1}
2 Related Work

Stance Detection Prior work on stance explored its connection to argument mining (Boltužić and Šnajder, 2014; Sobhani et al., 2015), opinion mining (Wang et al., 2019), and sentiment analysis (Mohammad et al., 2017; Aldayel and Magdy, 2019). Debating platforms were used as data source for stance (Somasundaran and Wiebe, 2010; Hasan and Ng, 2014; Aharoni et al., 2014), and more recently it was Twitter (Mohammad et al., 2016; Gorrell et al., 2019). With time, the definition of stance has become more nuanced (Küçük and Can, 2020), as well as its applications (Zubiaga et al., 2018; Hardalov et al., 2021b). Settings vary with respect to implicit (Hasan and Ng, 2013; Gorrell et al., 2019) or explicit topics (Augenstein et al., 2016; Stab et al., 2018; Allaway and McKeown, 2020), claims (Baly et al., 2018; Chen et al., 2019; Hanselowski et al., 2019; Conforti et al., 2020a,b) or headlines (Ferreira and Vlachos, 2016; Habernal et al., 2018; Mohtarami et al., 2018).

The focus, however, has been on homogeneous text, as opposed to cross-platform or cross-domain. Exceptions are Stab et al. (2018), who worked on heterogeneous text, but limited to eight topics, and Schiller et al. (2021), who combined datasets from different domains, but used in-domain multi-task learning, and Mohtarami et al. (2019) and Hardalov et al. (2021a), who used a cross-lingual setup. In contrast, we focus on cross-domain learning on 16 datasets, and out-of-domain evaluation.

Domain Adaptation Domain adaptation was studied in supervised settings, where in addition to the source-domain data, a (small) amount of labeled data in the target domain is also available (Daumé III, 2007; Finkel and Manning, 2009; Donahue et al., 2013; Yao et al., 2015; Mou et al., 2016; Lin and Lu, 2018), and in unsupervised settings, without labeled target-domain data (Blitzer et al., 2006; Lipton et al., 2018; Shah et al., 2018; Mohtarami et al., 2019; Bjerva et al., 2020; Wright and Augenstein, 2020). Recently, domain adaptation was applied to pre-trained Transformers (Lin et al., 2020). One direction therein are architectural changes (method-centric): Ma et al. (2019) proposed curriculum learning with domain-discriminative data selection, Wright and Augenstein (2020) investigated an unsupervised multi-source approach with Mixture of Experts and domain adversarial training (Ganin et al., 2016).

Another direction is data-centric adaptation: Han and Eisenstein (2019); Rietzler et al. (2020) used MLM fine-tuning on target-domain data. Gururangan et al. (2020) showed alternate domain-adaptive (in-domain data) and task-adaptive (out-of-domain unlabelled data) pre-training.

Label Embeddings Label embeddings can capture, in an unsupervised fashion, the complex relations between target labels for multiple datasets or tasks. They can boost the end-task performance for various deep learning architectures, e.g., CNNs (Zhang et al., 2018; Pappas and Henderson, 2019), RNNs (Augenstein et al., 2018, 2019), and Transformers (Chang et al., 2020). Recent work has proposed different perspectives for learning label embeddings: Beryozkin et al. (2019) trained a named entity recogniser from heterogeneous tag sets, Chai et al. (2020) used label descriptions for text classification, Rethmeier and Augenstein (2020) explored contrastive label embeddings for long-tail learning.

In our work, we propose an end-to-end framework to learn from heterogeneous labels based on unsupervised domain adaptation and label embeddings, and an unsupervised approach to obtain predictions for an unseen set of user-defined targets, using the similarity between label names.

3 Stance Detection Datasets

In this section, we provide a brief overview of the 16 stance datasets included in our study, and we show their key characteristics in Table 1. More details are given in Section 3.1 and in the Appendix (Section B.1). We further motivate the source groupings used in our experiments and analysis (Section 3.3).

3.1 Datasets

arc The Argument Reasoning Comprehension dataset has posts from the New York Times debate section on immigration and international affairs.

argmin The Argument Mining corpus presents arguments relevant to a particular topic from heterogeneous texts. Topics include controversial keywords like death penalty and gun control.

emergent The Emergent dataset is a collection of articles from rumour sites annotated by journalists.

fnc1 The Fake News Challenge dataset consists of news articles whose stance towards headlines is provided. It spans 300 topics from Emergent.

http://www.emergent.info/
### Dataset Characteristics

As is readily apparent from Table 1, the datasets differ based on the nature of the target and the context, as well as the stance labels.
The Target is the object of the stance. It can be a Claim, e.g., “Corporal punishment be used in K-12 schools.”, a Headline, e.g., “A meteorite landed in Nicaragua”, a Person, a Topic, e.g., abortion, healthcare, or None (i.e., an implicit target). Respectively, the Context, which is where the stance is expressed, can be an Article, a Claim, a Post, e.g., in a debate, a Thread, i.e., a chain of forum posts, a Sentence, or a Tweet. More examples from each dataset can be found in Table 7 in Appendix B.

Moreover, the diversity of the datasets is also reflected in their label names, ranging from different variants of positive, negative, and neutral to labels such as query or comment. The mapping between them is one of the core challenges we address, and it is discussed in more detail in Section 4.2.

Finally, the datasets differ in their size (see Table 2), varying from 800 to 75K examples. A complementary analysis of their quantitative characteristics, such as how the splits were chosen, the similarity between their training and testing parts, and their vocabularies, can be found in Appendix B.

### 3.3 Source Groups

Defining source groups/domains is an important part of this study, as they allow for better understanding of the relationship between datasets, which we leverage through domain-adaptive modelling (Section 4). Moreover, we use them to outline phenomena in the results that similar datasets share (Section 5). Table 1 shows these groupings.

Based on the aforementioned definitions of targets and context, we define the following groups: (i) Debates, (ii) News, (iii) Social Media, and (iv) Various.

We combine argmin (Web searches), ibmcs (Encyclopedia), and vast into Various, since they do not fit into any other group.

To demonstrate the feasibility of our groupings, we plot the 16 datasets in a latent vector space. We proportionally sample 25K examples, and we pass them through a RoBERTaBase (Liu et al., 2019) cased model without any training. The input has the following form: [CLS] context [SEP] target. Next, we take the [CLS] token representations, and we plot them in Figure 1 using tSNE (van der Maaten and Hinton, 2008). We can see that Social Media datasets are grouped top-right, Debates are in the middle, and News are on the left (except for Snopes). The Various datasets, ibmcs and argmin, are placed in between the aforementioned groups (i.e., Debates and News), and argmin is scattered into small clusters, confirming that they do not fit well into other source categories. Moreover, the figure reflects the strong connections between vast and arc, as well as between fnc1 and emergent, as the former is derived from the latter. Finally, the clusters are well-separated and do not overlap, which highlights the rich diversity of the datasets, each of which has its own definition of stance.

### 4 Method

We propose a novel end-to-end framework for cross-domain label-adaptive stance detection. Our architecture (see Figure 2) is based on input representations from a pre-trained language model, adapted to source domains using Mixture-of-Experts and domain adversarial training (Section 4.1). We further use self-adaptive output representations obtained via label embeddings, and unsupervised alignment between seen and unseen target labels for out-of-domain datasets.

Unlike previous work, we focus on learning the relationship between datasets and their label inventories in an unsupervised fashion. Moreover, our Mixture-of-Experts model is more compact than the one proposed by Wright and Augenstein (2020), as we introduce a parameter-efficient architecture with layers that are shared between the experts. Finally, we explore the capability of the model to predict from unseen user-defined target.

With this framework, we solve two main challenges: (i) training domain-adaptive models over a large number of datasets from a variety of source domains, and (ii) predicting an unseen label from a disjoint set of over 50 unique labels.
4.1 Cross-Domain Stance Detection

Mixture-of-Experts (MoE) is a well-known technique for multi-source domain adaptation (Guo et al., 2018; Li et al., 2018). Recently, this framework was extended to large pre-trained Transformers (Wright and Augenstein, 2020).

In particular, for each domain $k \in K$, there is a domain expert $f_k$, and a shared, global model $f_g$. We define $K = 4$, as we use four different domains (see Section 3.3), making this approach appealing to further encourage knowledge sharing between datasets. Further, the models produce a set of probabilities $p_k$ from each expert and $p_g$ from the global model, for all the (often shared) target labels. Then, the final output of the model is obtained by passing these predictions through a combination function, e.g., mean average, weighted average, or attention-based combination (Wright and Augenstein, 2020). We use mean average to gather the final distribution across the label space:

$$ p_A(x, K) = \frac{1}{|K| + 1} \sum_{k \in K} p_k(x) + p_g(x) $$

Mixture-of-Experts with Label Embeddings (MoLE) We propose several changes to Mixture-of-Experts to improve the model’s parameter efficiency, reduce training and inference times, and allow for different label inventories for each task.$^5$

First, in contrast to Wright and Augenstein (2020), we use a shared encoder, here, RoBERTa (Liu et al., 2019), instead of a separate large Transformer model for each domain. Next, for each domain expert and the global shared model, we add a single Transformer (Vaswani et al., 2017) layer on top of the encoder block. We thereby retain the domain experts while sharing information through the encoder. This approach reduces the number of parameters by a factor of the size of the entire model divided by the size of a single layer, i.e., we only use four additional layers (one such encoder block per domain) instead of 48 (the number of layers in RoBERTa$\text{Base}$, not counting embedding layers). For convenience, we set the hidden sizes in the newly-added blocks to match the encoder’s. Next, each domain expert receives as input the representations from the shared encoder of all tokens in the original sequence. Finally, we obtain a domain-specific and a global representation for the input sentence from the $[\text{CLS}]$ tokens. These hidden representations are denoted as $H \in \mathbb{R}^{K \times d_h}$, where $K$ is the number of domains, and $d_h$ is the model’s hidden size. They are passed through a single label embedding layer to obtain the probability distributions.

Domain-Adversarial training was introduced as part of the Domain-adversarial neural networks (DANN) (Ganin and Lempitsky, 2015). The aim is to learn a task classifier by confusing an auxiliary domain classifier optimised to predict a meta target, i.e., the domain of an example. This approach has shown promising results for many NLP applications (Li et al., 2018; Gui et al., 2017; Wright and Augenstein, 2020). Formally, it forces the model to learn domain-invariant representations, both for the source and for the target domains. The latter is done with an adversarial loss function, where we minimise the task objective $f_g$, and maximise the confusion in the domain classifier $f_d$ for an input sample $x$ (see Eq. 2). We implement this with a gradient reversal layer, which ensures that the source and the target domains are made to be similar.

$$ L_D = \max_{\theta_D} \min_{\theta_G} -d \log f_d(f_g(x)) $$

4.2 Cross-Domain Label-Adaptive Prediction

The second major challenge is how to obtain predictions for out-of-domain datasets. We want to emphasise that just a few of our 16 datasets share the same set of labels (see Section 3); yet, many labels in different datasets are semantically related.
Label Embeddings (LEL)  In multi-task learning, each task typically has its own task-specific labels (in our case, dataset-specific labels), which are predicted in a joint model using separate output layers. However, these dataset-specific labels are not entirely orthogonal to each other (see Section 3). Therefore, we adopt label embeddings to encourage the model to learn task relations in an unsupervised fashion using a common vector space. In particular, we add a Label Embeddings Layer, or LELs, (Augenstein et al., 2018), which learns a label compatibility function between the hidden representation of the input \( h \), here the one from the [CLS] token, and an embedding matrix \( L \):

\[
p = \text{softmax}(Lh)
\]

where \( L \in \mathbb{R}^{(\sum L_i) \times h} \) is the shared label embedding matrix for all datasets, and \( l \) is a hyperparameter for the dimensionality of each vector.

We set the size of the embeddings to match the hidden size of the model, and obtain the hidden representation \( h \) from the last layer of the pre-trained language model. Afterwards, we optimise a cross-entropy objective over all labels, masking the unrelated ones and keeping visible only the ones from the target datasets for a sample in the batch. We use the same masking procedure at inference time.

Label-Adaptive Prediction  In an unsupervised out-of-domain setting, there is no direct way to obtain a probability distribution over the set of test labels. Label embeddings are an easy indirect option for obtaining these predictions, as they can be used to measure the similarity between source and target labels. We investigate several alternatives.

Hard Mapping  A supervised option is to define a set of meta-groups (hard labels), here six, as shown in Table 3, then to train the model on these labels. E.g., if the out-of-domain dataset is snopes, then its labels are replaced with meta-group labels \(- agree \Rightarrow positive, and refute \Rightarrow negative\), and thus we can directly use the predictions from the model for out-of-domain datasets. However, this approach has several shortcomings: (i) labels have to be grouped manually, (ii) the meta-groups should be large enough to cover different task definitions, e.g. the dataset’s label inventory may vary in size, and, most importantly, (iii) any change in groupings would require full model re-training.

Soft Mapping  To overcome these limitations, we propose a simple, yet effective, entirely unsupervised procedure involving only the label names.

<table>
<thead>
<tr>
<th>Group</th>
<th>Task_labels Included</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>arc_agree, argmin_argument_for, emergent_for, fn1_agree, iac1_pro, mtsd_favor, perspectrum_support, poldeb_for, rumor_endorse, sdm_for, semeval20166_favor, semeval20197_support, snopes_agree, vast_pro, wtwt_support</td>
</tr>
<tr>
<td>Negative</td>
<td>arc_disagree, argmin_argument_against, emergent_against, fn1_disagree, iac1_ant, ibmcsc_con, mtsd_against, perspectrum_answer, poldeb_disagree, rumor_deny, sdm_against, semeval20166_against, semeval20197_deny, snopes_refute, vast_con, wtwt_refute</td>
</tr>
<tr>
<td>Discuss</td>
<td>arc_discuss, emergent_observing, fn1_discuss, rumor_question, semeval20197_query, wtwt_comment</td>
</tr>
<tr>
<td>Other</td>
<td>arc_unrelated, fn1_unrelated, iac1_other, mtsd_none, rumor_unrelated, semeval20197_comment, wtwt_unrelated</td>
</tr>
<tr>
<td>Neutral</td>
<td>rumor_neutral, vast_neutral</td>
</tr>
</tbody>
</table>

Table 3: Hard mapping of labels to categories.

More precisely, we measure the similarity between the names of the labels across datasets. This is an intuitive approach for finding a matching label without further context, e.g., for is probably close to agree, and refute is close to against. In particular, given a set of out-of-domain target labels \( Y^t \in \{y_1^t, \ldots, y_k^t\} \), and a set of predictions from in-domain labels \( P^h \in \{p_1^h, \ldots, p_m^h\} \), \( P_i^h \in \{y_1^i, \ldots, y_j^i\} \), we select the label from \( Y^t \) with the highest cosine similarity to the predicted label \( p_i^h \):

\[
p_i^h = \arg\max_{y^t \in Y^t} \cos(y^t, P_i^h)
\]

where \( k \) is the number of out-of-domain labels, \( m \) the number of out-of-domain examples, and \( j \) the number of in-domain labels. The procedure can generalise to any labels, without the need for additional supervision. To illustrate this, the embedding spaces of pre-trained embedding models for our 16 datasets are visualised in Appendix C.

Weak Mapping  Nevertheless, as proposed, this procedure only takes label names into account, in contrast to the hard labels that rely on human expertise. This makes combining the labels in a weakly supervised manner an appealing alternative. For this, we measure label similarities as proposed, but incorporate some supervision for defining the embeddings. We first group the labels into six separate categories to define their nearest neighbours (see Table 3). Then, we choose the most similar label for the target domain from these neighbours.
Table 4: Label grouping and the closest neighbours of each, sorted from closest to most distant.

<table>
<thead>
<tr>
<th>Group</th>
<th>Closest Neighbours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>Other, Neutral, Discuss, Negative</td>
</tr>
<tr>
<td>Other</td>
<td>Neutral, Discuss, Positive, Negative</td>
</tr>
<tr>
<td>Neutral</td>
<td>Discuss, Other, Positive, Negative</td>
</tr>
<tr>
<td>Discuss</td>
<td>Neutral, Other, Negative, Positive</td>
</tr>
<tr>
<td>Negative</td>
<td>Discuss, Neutral, Other, Positive</td>
</tr>
</tbody>
</table>

The list of neighbours is defined by the group of the predicted label. However, there is no guarantee that there will be a match for the target domain within the same group, and thus we further define group-level neighbourhoods (see Table 4), as it is not feasible to define the neighbours for all (more than 50) labels individually. One drawback is that each new label/group must define a neighbourhood with similar labels – and vice-versa, it should be assigned a position in the neighbourhoods of the existing labels.

4.3 Training

We train the model using the following loss:

\[
L_s = \frac{1}{N} \sum_i y_i \log p_X(x, S') \quad (5)
\]

\[
L_t = \frac{1}{N} \sum_i y_i \log p_t(x) \quad (6)
\]

\[
L = \lambda L_s + (1 - \lambda) L_t + \gamma L_D \quad (7)
\]

First, we sum the source-domain loss \(L_s\) with the meta-target loss from the domain expert sub-network \(L_t\), where the contribution of each is balanced by a single hyper-parameter \(\lambda\), set to 0.5. Next, we add the domain adversarial loss \(L_D\), and we multiply it by a weighting factor \(\gamma\), which is set to a small positive number to prevent this regulariser from dominating the overall loss. We set \(\gamma\) to 0.01. Furthermore, since our dataset is quite diverse even in the four source domains that we outlined, we optimise the domain-adaptive loss towards a meta-class for each dataset, instead of the domain.

5 Experiments

We consider three evaluation setups: (i) no training, random and majority class baselines; (ii) in-domain, training then testing on all datasets; and (iii) out-of-domain, i.e., leave-one-dataset-out training for all datasets. The reported per-dataset scores are macro-averaged \(F_1\), which are additionally averaged to obtain per-experiment scores.

5.1 Baselines

Majority class baseline calculated from the distributions of the labels in each test set.

Random baseline Each test instance is assigned a target label at random with equal probability.

Logistic Regression A logistic regression trained using TF.IDF word unigrams. The input is a concatenation of the target and context vectors.

Multi-task learning (MTL) A single projection layer for each dataset is added on top of a pre-trained language model (BERT (Devlin et al., 2019) or RoBERTa (Liu et al., 2019)). We then pass the [CLS] token representations through the dataset-specific layer. Finally, we propagate the errors only through that layer (and the base model), without updating parameters for other datasets.

5.2 Evaluation Results

In-Domain Experiments We train and test on all datasets; the results are in Table 5. First, to find the best base model and set a baseline for MoLE, we evaluate two strong models: BERTBase uncased (Devlin et al., 2019), and RoBERTaBase cased\(^6\) (Liu et al., 2019). On our 16 datasets, RoBERTa outperforms BERT by 2 \(F_1\) points absolute on average.

In the following rows of Table 5, we show results for our model (MoLE), i.e., Mixture of Experts with Label Embeddings and Domain-Adversarial Training (see Section 4). Its full version scores the highest in terms of \(F_1 = 65.55\), which is 0.43 absolute points better than the MTL (RoBERTaBase) baseline. In particular, it outperforms this strong baseline on nine of the 16 datasets. Nevertheless, neither MoLE, nor any of its variations improves the results for mtsd, rumor, and semeval2019t6 over the MTL (RoBERTaBase) model. We attribute this to their specifics: mtsd is the only dataset where the target is a Person, rumor and semeval2019t6 both focus on stance towards rumors, but the data for rumor is from 2009–2011, and semeval2019t6 has an implicit target.

Next, we present ablations – we sequentially remove a prominent component from the proposed model (MoLE). First, we optimise the model without the domain-adversarial loss. Removing the DANN leads to worse results on ten of the datasets, and a drop in the average \(F_1\) compared to MoLE.

\(^6\)We choose the uncased version of BERT due to its wide use in similar tasks; RoBERTa is cased by nature. We use the base versions of the models for computational efficiency.
We further evaluate all mapping procedures proposed in Section 4.2 for out-of-domain prediction: (i) hard, (ii) weak, and (iii) soft mapping.

The hard mapping approach outperforms the majority class baseline, but it falls almost 3 points absolute short compared to the random baseline, while failing to do better than random on more than half of the datasets. The two main factors for this are that (i) the predictions are dominated by the meta-targets with the most examples, i.e., *discuss*, (ii) the model struggles to converge on the training set, due to diversity in the datasets and their labels.

The weak and the soft embeddings share the same set of predictions, as their training procedure is the same – the only difference between them are the embeddings used to align the prediction to the set of unseen targets. The weak mappings achieve the highest average F1 among the out-of-domain models. For context, note that it is still 16% behind the best in-domain model. Furthermore, in this setup, we see that emergent scores 82%, just few points below the in-domain result – we suspect that this is due to the good alignment of labels with *fnc1*, as the two datasets are closely related.

For the soft mappings, we evaluate five well-established embedding models, i.e., *fastText* (Joulin et al., 2017), *GloVe* (Pennington et al., 2014), *RoBERTaBase*, and two sentiment-enriched ones, i.e. *sentiment-specific word embedding* (sswe, Tang et al. (2014)), and *RoBERTa Twitter Sentiment* (roberta-sentiment, Barbieri et al. (2020)). Our motivation for including the latter is that the names of the stance labels are often sentiment-related, and thus encoding that information into the latent space might yield better groupings (see Appendix C).

### Table 5: In-domain experiments. Results are shown in terms of F1. In the rows below MoLE, we remove (−) the components sequentially from it.

<table>
<thead>
<tr>
<th>MoLE w/ Hard Mapping</th>
<th>32.78</th>
<th>25.39</th>
<th>35.15</th>
<th>29.55</th>
<th>22.80</th>
<th>16.13</th>
<th>58.49</th>
<th>47.05</th>
<th>29.28</th>
<th>23.34</th>
<th>32.93</th>
<th>37.01</th>
<th>21.85</th>
<th>16.10</th>
<th>34.16</th>
<th>72.93</th>
<th>22.89</th>
</tr>
</thead>
<tbody>
<tr>
<td>MoLE w/ Weak Mapping</td>
<td>49.20</td>
<td>51.81</td>
<td>38.97</td>
<td>58.48</td>
<td>47.23</td>
<td>53.96</td>
<td>82.07</td>
<td>51.57</td>
<td>56.97</td>
<td>40.13</td>
<td>51.29</td>
<td>36.31</td>
<td>31.75</td>
<td>22.78</td>
<td>50.71</td>
<td>75.69</td>
<td>37.15</td>
</tr>
<tr>
<td>MoLE w/ Soft Mapping</td>
<td>42.67</td>
<td>48.31</td>
<td>31.23</td>
<td>62.73</td>
<td>54.19</td>
<td>49.58</td>
<td>46.86</td>
<td>53.46</td>
<td>53.58</td>
<td>37.88</td>
<td>44.38</td>
<td>36.77</td>
<td>24.40</td>
<td>21.53</td>
<td>56.48</td>
<td>59.26</td>
<td>19.67</td>
</tr>
<tr>
<td>w/ fasttext</td>
<td>39.00</td>
<td>46.54</td>
<td>9.32</td>
<td>48.87</td>
<td>52.20</td>
<td>51.97</td>
<td>40.32</td>
<td>48.36</td>
<td>49.32</td>
<td>34.38</td>
<td>44.42</td>
<td>24.07</td>
<td>7.68</td>
<td>28.97</td>
<td>57.78</td>
<td>59.14</td>
<td>19.80</td>
</tr>
<tr>
<td>w/ glove</td>
<td>32.22</td>
<td>44.88</td>
<td>32.12</td>
<td>36.14</td>
<td>39.38</td>
<td>31.24</td>
<td>23.02</td>
<td>33.07</td>
<td>49.60</td>
<td>53.84</td>
<td>12.10</td>
<td>17.76</td>
<td>6.97</td>
<td>25.51</td>
<td>33.90</td>
<td>65.32</td>
<td>30.96</td>
</tr>
<tr>
<td>w/ roberta-base</td>
<td>37.06</td>
<td>44.81</td>
<td>26.67</td>
<td>35.18</td>
<td>50.69</td>
<td>50.65</td>
<td>19.55</td>
<td>42.75</td>
<td>45.94</td>
<td>28.65</td>
<td>15.66</td>
<td>23.25</td>
<td>28.92</td>
<td>24.64</td>
<td>55.90</td>
<td>72.11</td>
<td>28.05</td>
</tr>
<tr>
<td>w/ roberta-sentiment</td>
<td>37.10</td>
<td>45.11</td>
<td>23.80</td>
<td>36.14</td>
<td>45.73</td>
<td>51.23</td>
<td>38.30</td>
<td>57.31</td>
<td>43.93</td>
<td>28.97</td>
<td>9.14</td>
<td>34.02</td>
<td>6.38</td>
<td>21.18</td>
<td>57.26</td>
<td>60.03</td>
<td>24.31</td>
</tr>
</tbody>
</table>

### Table 6: Out-of-domain experiments. Results are shown in terms of F1.

<table>
<thead>
<tr>
<th>Favg</th>
<th>arc</th>
<th>iacl</th>
<th>perspectum</th>
<th>poldebl</th>
<th>scl</th>
<th>emergent</th>
<th>fnc1</th>
<th>snopes</th>
<th>mtld</th>
<th>rumor</th>
<th>semeval16</th>
<th>semeval19</th>
<th>wtwt</th>
<th>argmin</th>
<th>ibmcs</th>
<th>vast</th>
</tr>
</thead>
<tbody>
<tr>
<td>Majority class baseline</td>
<td>27.60</td>
<td>21.45</td>
<td>21.27</td>
<td>34.66</td>
<td>39.38</td>
<td>35.30</td>
<td>21.30</td>
<td>20.96</td>
<td>43.98</td>
<td>19.49</td>
<td>25.15</td>
<td>24.27</td>
<td>22.34</td>
<td>15.91</td>
<td>33.83</td>
<td>34.06</td>
</tr>
<tr>
<td>Random baseline</td>
<td>35.19</td>
<td>15.80</td>
<td>30.66</td>
<td>30.06</td>
<td>48.67</td>
<td>50.08</td>
<td>31.83</td>
<td>18.64</td>
<td>45.49</td>
<td>33.15</td>
<td>20.43</td>
<td>31.11</td>
<td>17.02</td>
<td>20.01</td>
<td>49.94</td>
<td>50.08</td>
</tr>
<tr>
<td>Logistic Regression</td>
<td>41.35</td>
<td>21.43</td>
<td>28.68</td>
<td>61.33</td>
<td>72.30</td>
<td>44.63</td>
<td>61.30</td>
<td>24.02</td>
<td>59.32</td>
<td>44.29</td>
<td>21.93</td>
<td>49.82</td>
<td>22.34</td>
<td>32.32</td>
<td>51.06</td>
<td>37.13</td>
</tr>
<tr>
<td>MTL w/ BERTbase</td>
<td>63.11</td>
<td>63.19</td>
<td>45.30</td>
<td>78.62</td>
<td>50.76</td>
<td>64.03</td>
<td>86.23</td>
<td>74.48</td>
<td>71.55</td>
<td>56.36</td>
<td>60.26</td>
<td>68.28</td>
<td>61.03</td>
<td>63.59</td>
<td>59.05</td>
<td>68.55</td>
</tr>
<tr>
<td>MTL w/ RoBERTabase</td>
<td>65.12</td>
<td>64.52</td>
<td>35.77</td>
<td>82.38</td>
<td>53.83</td>
<td>59.43</td>
<td>83.91</td>
<td>75.29</td>
<td>74.95</td>
<td>65.87</td>
<td>71.23</td>
<td>70.46</td>
<td>59.42</td>
<td>67.64</td>
<td>61.79</td>
<td>77.27</td>
</tr>
<tr>
<td>MoLE (Our Model)</td>
<td>65.55</td>
<td>63.17</td>
<td>38.50</td>
<td>85.27</td>
<td>50.76</td>
<td>65.91</td>
<td>83.74</td>
<td>75.82</td>
<td>75.97</td>
<td>65.08</td>
<td>67.24</td>
<td>70.05</td>
<td>57.76</td>
<td>68.37</td>
<td>64.73</td>
<td>79.38</td>
</tr>
<tr>
<td>MoLE</td>
<td>65.40</td>
<td>64.28</td>
<td>37.20</td>
<td>83.38</td>
<td>53.99</td>
<td>62.79</td>
<td>83.44</td>
<td>75.47</td>
<td>74.77</td>
<td>65.44</td>
<td>70.41</td>
<td>72.08</td>
<td>54.68</td>
<td>68.90</td>
<td>62.29</td>
<td>78.42</td>
</tr>
<tr>
<td>w/ MoE</td>
<td>64.68</td>
<td>65.18</td>
<td>38.41</td>
<td>81.46</td>
<td>51.34</td>
<td>64.57</td>
<td>84.60</td>
<td>75.79</td>
<td>74.05</td>
<td>65.69</td>
<td>61.07</td>
<td>69.99</td>
<td>56.67</td>
<td>69.03</td>
<td>62.25</td>
<td>76.87</td>
</tr>
</tbody>
</table>

However, this model does better in terms of points absolute on arc (1%), poldebl (3%), rumor (3%), and *semeval2016t7* (+2%). We attribute that to the more specialised domain representations being helpful, as some of the other datasets we trained on are very similar to those, e.g., *vast* is derived from arc. Moreover, removing domain adversarial training has a negative impact on the datasets with source *Various* (i.e., *argmin*, *ibmcs*, *vast*).

Clearly, forcing similar representations aids knowledge sharing among domain experts, as they score between 0.7 and 1.5 F1 lower compared to MoLE, the same behaviour as observed in other ablations. The last row of Table 5 (−MoE) shows results for RoBERTabase with Label Embeddings. It performs the worst of all RoBERTa-based models, scoring 0.5 points lower than MTL overall. Note that it is not possible to present results for a MoE-based model without Label Embeddings, due to the discrepancy in the label inventories, both between and within domains, which means a standard voting procedure cannot be applied (see Section 4.2).
Figure 3: Pearson correlation between the out-of-domain results for our model (MoLE w/ weak mappings) in terms of F1 score and dataset characteristics. The (type) of the features is shown in parenthesis.

We examine the performance of soft mapping w/ fastText in more detail as they score the highest among other strong alternatives. Interestingly, the soft mappings benefit from splitting the predictions for the labels in the same group, such as \texttt{wtwt__comment} and all discuss-related, which leads to the better performance on perspersion, poldeb, fncl, argmin in comparison to the weak mappings. Nevertheless, this also introduces some errors. An illustrative example are short words – anti, pro, con, which are distant from all other label names in our pool (see Figure 5 in Appendix C for an illustration). The neighbourhoods are sometimes hard to interpret, e.g., con is not the closest word for any predicted labels in vast, and is aligned only with undermine, unrelated in ibmcs.

6 Discussion

We further study the correlation between the scores for the best model in the out-of-domain setup MoLE w/ weak mappings and a rich set of quantitative and stance-related characteristics of the datasets (these are further discussed in Section 3 and in Appendix B). In particular, we represent each dataset as a set of features, e.g., \texttt{fncl} would have target – News, training set size of 42,476, etc., and then we measure the Pearson correlation between these features and the model’s F1 scores per dataset.

Figure 3 shows the most important factors for out-of-domain performance.\(^7\) We see positive correlations of F1 with the training, and the development set sizes, and a negative one with the testing set size, which suggests that large datasets are indeed harder for the model. Interestingly, if there is an overlap in the targets between the testing and the training sets, the model’s F1 is worse; however, this is not true for context overlap. Unsurprisingly, the size of the vocabulary is a factor that negatively impacts F1, and its moderate negative correlation with the model’s scores confirms that.

The domain, the target and the context types are also important facets: the News domain has a sizable positive correlation with F1, which is also true for the related features Headline target and Article body. Another positive correlation is for having a Claim as the context. On the contrary, a key factor that hinders model performance is Social Media text, i.e., having a tweet as a context.

7 Conclusion and Future Work

We have proposed a novel end-to-end unsupervised framework for out-of-domain stance detection with respect to unseen labels. In particular, we combined domain adaptation techniques such as Mixture-of-Experts and domain-adversarial training with label embeddings, which yielded sizable performance gains on 16 datasets over strong baselines: both in-domain, i.e., for seen targets, and out-of-domain, i.e., for unseen targets. Moreover, we performed an exhaustive analysis of the cross-domain results, and we highlighted the most important factors influencing the model performance.

In future work, we plan to experiment with more datasets, including non-English ones, as well as with other formulations of the stance detection task, e.g., stance of a person (Darwish et al., 2020) or of a news medium (Stefanov et al., 2020) with respect to a claim or a topic.

Acknowledgments

We thank the anonymous reviewers for their helpful questions and comments, which have helped us improve the quality of the paper.

We also would like to thank Guillaume Bouchard for the useful feedback. Finally, we thank the authors of the stance datasets for open-sourcing and providing us with their data.

\(^7\) Some of the factors in the analysis are not independent, e.g., Social Media as a domain, and Tweet as a context.
**Ethics and Broader Impact**

**Dataset Collection**  We use publicly available datasets and we have no control over the way they were collected. For datasets that distributed their data as Twitter IDs, we used the Twitter API\(^8\) to obtain the full text of the tweets, which is in accordance with the terms of use outlined by Twitter.\(^9\) Note that we only downloaded public tweets.

**Biases**  We note that some of the annotations are subjective. Thus, it is inevitable that there would be certain biases in the datasets. These biases, in turn, will likely be exacerbated by the supervised models trained on them (Waseem et al., 2021). This is beyond our control, as are the potential biases in pre-trained large-scale transformers such as BERT and RoBERTa, which we use in our experiments.

**Intended Use and Misuse Potential**  Our models can enable analysis of text and social media content, which could be of interest to business, to fact-checkers, journalists, social media platforms, and policymakers. However, they could also be misused by malicious actors, especially as most of the datasets we consider in this paper are obtained from social media. Most datasets compiled from social media present some risk of misuse. We, therefore, ask researchers to exercise caution.

**Environmental Impact**  We would also like to note that the use of large-scale Transformers requires a lot of computations and the use of GPUs/TPUs for training, which contributes to global warming (Strubell et al., 2019). This is a bit less of an issue in our case, as we do not train such models from scratch; rather, we fine-tune them on relatively small datasets. Moreover, running on a CPU for inference, once the model has been fine-tuned, is perfectly feasible, and CPUs contribute much less to global warming.

**References**


Genady Beryozkin, Yoel Drori, Oren Gilon, Tzvika Hartman, and Idan Szpektor. 2019. *A joint named-entity recognizer for heterogeneous tag-sets using a 1

---

\(^8\)http://developer.twitter.com/en/docs


Filip Boltužić and Jan Šnajder. 2014. Back up your stance: Recognizing arguments in online discussions. In Proceedings of the First Workshop on Argumentation Mining, ArgMining ’14, pages 49–58, Baltimore, Maryland, USA.


9021


Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for Twitter sentiment classification. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL ’14, pages 1555–1565, Baltimore, Maryland, USA.


A Fine-Tuning and Hyper-Parameters

• All models are developed in Python using PyTorch (Paszke et al., 2019) and the Transformers library (Wolf et al., 2020).

• All models use Adam (Kingma and Ba, 2015) with weight decay 1e-8, $\beta_1$ 0.9, $\beta_2$ 0.999, $\epsilon$ 1e-08, and are trained for five epochs with batch size 64, and maximum length of 100 tokens.\textsuperscript{10}

• RoBERTa (Liu et al., 2019) is trained w/ LR 1e-05, warmup 0.06, BERT (Devlin et al., 2019) is trained w/ LR 3e-05, warmup 0.1.

• The values of the hyper-parameters were selected on the development set.

• We chose the best model checkpoint based on the performance on the development set.

• For the MTL/MoE models, we sampled each batch from a single randomly selected dataset/domain.

• We used the same seed for all experiments.

• Each experiment took around 1h 15m on a single NVIDIA V100 GPU using half precision.

• For logistic regression, we converted the text to lowercase, removed the stop words, and limited the dictionary in the TF.IDF to 15,000 unigrams. We built the vocabulary using the concatenated target and context. The target and the context were transformed separately and concatenated to form the input vector.

B Dataset Analysis

B.1 Data Splits

Here, we provide more detail about the splits we used for the datasets, in cases where there is a deviation from the original. For the datasets in common, we used the splitting by Schiller et al. (2021). We further tried to enforce a larger domain diversity between the training, the development, and the testing sets; hereby, we put (whenever possible) all examples from a particular topic (domain) strictly into a single split.

**argmin** We removed all non-arguments. The training, the development, and the test data splits consist of five, one, and two topics, respectively.

**iac1** Split with no intersection of topics between the training, the development, and the testing sets.

**ibmcs** Pre-defined training and testing splits. We further reserved 10% of the training data for development set.

**mtsd** We used the pre-defined splits, but we created two pairs for each example: a positive and a negative one with respect to the target.

**poledb** We used the domains Healthcare, Guns, Gay Rights and God for training, Abortion for development, and Creation for testing.

**rumor** We used the airfrance rumour for our test set, and we split the remaining data in ratio 9:1 for training and development, respectively.

**wtwt** We used DIS_FOXA operation for testing, AET_HUM for development, and the rest for training. To standardize the targets, we rewrote them as sentences, i.e., company X acquires company Y.

**scd** We used a split with Marijuana for development, Obama for testing, and the rest for training.

**semeval2016t6** We split it to increase the size of the development set.

**snopes** We adjusted the splits for compatibility with the stance setup. We further extracted and converted the rumours and their evidence into target–context pairs.

B.2 Overlap Statistics

Next, in Table 8, we examine the proportion of contexts and targets from the development and the testing datasets that are also present in the training split. We did not change the original data in any way, and we used the splits as described in Section B.1.
Table 8 further shows statistics about the datasets in terms of the number of words and sub-words they contain (see Table 9). The first column in the table shows the number of unique tokens (word types) in each dataset after tokenisation using NLTK’s casual tokeniser (Loper and Bird, 2002), which retains the casing of the words; thus word types of different casing are counted separately. We observe that the datasets with the largest vocabularies are those (i) with higher numbers of examples (fnc1 and wtwt), (ii) whose contexts are threads rather than single posts (iac1 has over 1,300 words on average), and (iii) that cover diverse topics such as poldeb with six unrelated ones. In contrast, small or narrow datasets such as ibmcs have the smallest vocabularies (fewer than 5,000 words).

In the subsequent columns, we report statistics in terms of number of sub-words (i.e., SentencePieces (Kudo and Richardson, 2018) from RoBERTaBase’s tokeniser). With that, we want to present the expected coverage in terms of tokens for a pre-trained model. On average, most of the datasets are well under 100 tokens in length, which is commonly observed for tweets, but some datasets have a higher average number of tokens, e.g., debate-related datasets such as arc, poldeb, scd, vast fit in 200 tokens on average, which is also the case for datasets containing large news articles or use social media threads as context (fnc1, iac1), where the average length is over 500.
Figure 4: Proportion of word types in the dataset in row \(i\) that are also present in the dataset in column \(j\) (stop-words removed, and case preserved).

### Table 8: Percentage of overlap between development/testing and training data. The (T)arget and the (C)ontext columns show the overlap in the respective individual fields; (F)ull shows the overall overlap.

<table>
<thead>
<tr>
<th>% of split in Train</th>
<th>Dev</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>arc</td>
<td>1.9</td>
<td>100.0</td>
</tr>
<tr>
<td>iac1</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>perspectrum</td>
<td>1.5</td>
<td>1.5</td>
</tr>
<tr>
<td>poldeb</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>scd</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>emergent</td>
<td>1.7</td>
<td>100.0</td>
</tr>
<tr>
<td>fnl</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>rumor</td>
<td>0.8</td>
<td>100.0</td>
</tr>
<tr>
<td>semeval20166</td>
<td>17.6</td>
<td>100.0</td>
</tr>
<tr>
<td>semeval20197</td>
<td>0.0</td>
<td>100.0</td>
</tr>
<tr>
<td>wtt</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>argmin</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>ibmcs</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>vast</td>
<td>0.0</td>
<td>43.6</td>
</tr>
</tbody>
</table>

Table 9: Statistics about the sub-word tokens for each dataset (using the RoBERTa\textsubscript{Base} tokeniser). The numbers in parenthesis show the word counts after the NLTK tokeniser was used.

<table>
<thead>
<tr>
<th>Tokenization</th>
<th>Words</th>
<th>Tokens</th>
<th>25%</th>
<th>Median</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Words Unique</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>arc</td>
<td>27,855</td>
<td>126.0</td>
<td>118.3</td>
<td>84</td>
<td>116</td>
</tr>
<tr>
<td>argmin</td>
<td>17,990</td>
<td>30.6</td>
<td>28.6</td>
<td>20</td>
<td>28</td>
</tr>
<tr>
<td>emergent</td>
<td>6,940</td>
<td>27.6</td>
<td>23.1</td>
<td>22</td>
<td>26</td>
</tr>
<tr>
<td>fnl</td>
<td>40,738</td>
<td>503.2</td>
<td>432.2</td>
<td>279</td>
<td>413</td>
</tr>
<tr>
<td>iac1</td>
<td>88,478</td>
<td>1,554.7</td>
<td>1,347.9</td>
<td>132</td>
<td>390</td>
</tr>
<tr>
<td>ibmcs</td>
<td>5,007</td>
<td>23.4</td>
<td>21.9</td>
<td>18</td>
<td>23</td>
</tr>
<tr>
<td>mtd</td>
<td>9,799</td>
<td>36.3</td>
<td>25.9</td>
<td>33</td>
<td>36</td>
</tr>
<tr>
<td>perspectrum</td>
<td>9,999</td>
<td>22.4</td>
<td>20.4</td>
<td>17</td>
<td>21</td>
</tr>
<tr>
<td>poldeb</td>
<td>40,422</td>
<td>178.0</td>
<td>160.6</td>
<td>55</td>
<td>112</td>
</tr>
<tr>
<td>rumor</td>
<td>15,801</td>
<td>38.4</td>
<td>24.4</td>
<td>32</td>
<td>39</td>
</tr>
<tr>
<td>scd</td>
<td>23,592</td>
<td>151.4</td>
<td>134.8</td>
<td>39</td>
<td>78</td>
</tr>
<tr>
<td>semeval20166</td>
<td>15,016</td>
<td>32.8</td>
<td>22.1</td>
<td>28</td>
<td>33</td>
</tr>
<tr>
<td>semeval20197</td>
<td>20,789</td>
<td>33.5</td>
<td>22.0</td>
<td>17</td>
<td>27</td>
</tr>
<tr>
<td>wtt</td>
<td>33,896</td>
<td>53.4</td>
<td>45.5</td>
<td>40</td>
<td>51</td>
</tr>
<tr>
<td>vast</td>
<td>24,644</td>
<td>123.2</td>
<td>115.7</td>
<td>80</td>
<td>114</td>
</tr>
<tr>
<td>wtwt</td>
<td>102,672</td>
<td>45.7</td>
<td>23.1</td>
<td>39</td>
<td>46</td>
</tr>
</tbody>
</table>

Finally, Figure 4 shows the relative word overlap between datasets. The numbers in each cell show how much of the word types in dataset \(i\) (row) are contained in the dataset \(j\) (column). For example, in the first column in the last row (\(vast \Rightarrow arc\)), we see that 97% of the words in \(vast\) are also present in \(arc\). Similarly, in the first row and the last column, we see that 86% of the words in \(arc\) are also in \(vast\). Note that we sort the columns and the rows by their sources (see Table 1). We can see that datasets with the largest vocabularies (\(iac1\) and \(wtwt\)) have low overlap with other datasets, including with each other, up to 28% only (row-wise).

When looking at how many words in other datasets are contained in them (column-wise), we see that \(iac1\) has 50% or more vocabulary overlap with the other datasets, even with ones from different sources. Then, \(wtwt\)’s overlaps are 30–70%, which is expected as its texts are from social media and cover a single topic (company acquisitions). For datasets that are either small or cover few topics (\(emergent, ibmcs, perspectrum\)), we see that moderate to large part of their vocabularies is contained in other datasets; yet, the opposite is not true. Moreover, social media datasets are orthogonal to each other, with cross-overlaps of up to 50% (both row-wise and column-wise), except for \(wtwt\). This is also seen when measuring how much of other datasets’ vocabulary they contain (column-wise).
Here, we present the embeddings based on the labels’ names. We explore five sets of embeddings: (i) well-established ones such as fastText (Joulin et al., 2017), GloVe (Pennington et al., 2014), and RoBERTa (Liu et al., 2019), and (ii) sentiment-enriched ones like Sentiment-specific word embedding (SSWE) (Tang et al., 2014) and RoBERTa Twitter Sentiment (roberta-sentiment) (Barbieri et al., 2020). Our motivation for including the latter is that the names of the stance labels are often sentiment-related, and thus encoding that information into the latent space could yield better grouping. We encode each label with the corresponding word from the embedding’s directory for non-contextualized embeddings, if the name contains multiple words, e.g., argument for, then we split on white space, and we take the average of each word’s embeddings. For RoBERTa-based models, we take the representation from the [CLS] token. Finally, we project the obtained vectors into two dimensions using PCA. The resulting plots are shown in Figure 5.

Figure 5: Embedding spaces of the label names’ representations (PCA) from different embedding models.

C Embedding Label Spaces

Here, we present the embeddings based on the labels’ names. We explore five sets of embeddings: (i) well-established ones such as fastText (Joulin et al., 2017), GloVe (Pennington et al., 2014), and RoBERTa (Liu et al., 2019), and (ii) sentiment-enriched ones like Sentiment-specific word embedding (SSWE) (Tang et al., 2014) and RoBERTa Twitter Sentiment (roberta-sentiment) (Barbieri et al., 2020). Our motivation for including the latter is that the names of the stance labels are often sentiment-related, and thus encoding that information into the latent space could yield better grouping. We encode each label with the corresponding word from the embedding’s directory for non-contextualized embeddings, if the name contains multiple words, e.g., argument for, then we split on white space, and we take the average of each word’s embeddings. For RoBERTa-based models, we take the representation from the [CLS] token. Finally, we project the obtained vectors into two dimensions using PCA. The resulting plots are shown in Figure 5.