Syntopical Graphs for Computational Argumentation Tasks

Joe Barrow[*]
University of Maryland
jdbarrow@cs.umd.edu

Rajiv Jain
Adobe Research
rajijain@adobe.com

Nedim Lipka
Adobe Research
lipka@adobe.com

Franck Dernoncourt
Adobe Research
dernonco@adobe.com

Vlad I. Morariu
Adobe Research
morariu@adobe.com

Varun Manjunatha
Adobe Research
vmanjuna@adobe.com

Douglas W. Oard
University of Maryland
oard@umd.edu

Philip Resnik
University of Maryland
resnik@umd.edu

Henning Wachsmuth
Paderborn University
henningw@upb.de

Abstract

Approaches to computational argumentation tasks such as stance detection and aspect detection have largely focused on the text of individual claims, losing out on potentially valuable context from the broader collection of text. We present a general approach to these tasks motivated by syntopical reading, a reading process that emphasizes comparing and contrasting viewpoints in order to improve topic understanding. To capture collection-level context, we introduce the *syntopical graph*, a data structure for linking claims within a collection. A syntopical graph is a typed multi-graph where nodes represent claims and edges represent different possible pairwise relationships, such as entailment, paraphrase, or support. Experiments applying syntopical graphs to stance detection and aspect detection demonstrate state-of-the-art performance in each domain, significantly outperforming approaches that do not utilize collection-level information.

1 Introduction

Collections of text about the same topic such as news articles and research reports often present a variety of viewpoints. Adler and Van Doren (1940) proposed a formalized manual process for understanding a topic based on multiple viewpoints in their book, *How to Read a Book*, applying dialectics to collection browsing. This process consists of four levels of reading, the highest of which is **syntopical reading**. Syntopical reading is focused on understanding a core concept by reading a collection of works. It requires finding passages on the core concept that agree or disagree with each other, defining the issues, and analyzing the discussion to gain a better understanding of the core concept. The goal of the paper at hand is to operationalize the syntopical reading process computationally in order to help individuals make sense of a collection of documents for a given topic.

Viewed through the lens of computational argumentation, these documents state **claims** or conclusions that can be grouped by the **aspects** of the **topic** they discuss as well as by the **stance** they convey towards the topic (Stede and Schneider, 2018). An individual aiming to form a thorough understanding of the topic needs to get an overview of these viewpoints and their interactions. This may be hard even if adequate tool support for browsing the collection is available (Wachsmuth et al., 2017a; Stab et al., 2018; Chen et al., 2019).

We seek to enable systems that are capable of reconstructing **viewpoints** within a collection, where a viewpoint is expressed as a triple $V = (\text{topic}, \text{aspect}, \text{stance})$. We consider the argumentative unit of a claim to be the minimal expression of a viewpoint in natural language, such that a single viewpoint can have many claims expressing it. As an example, consider the following two claims:

> *"Nuclear energy emits zero CO_2."*
>
> *"Nuclear can provide a clean baseload, eliminating the need for fracking and coal mining."*

Within a collection these claims express:

$$V = (\textit{Nuclear Energy}, \textit{env. impact}, \text{PRO})$$

The goal of the systems we envision is thus to identify, group, and summarize the latent view-

[*] Work done while interning at Adobe Research.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 1583–1595
August 1–6, 2021.

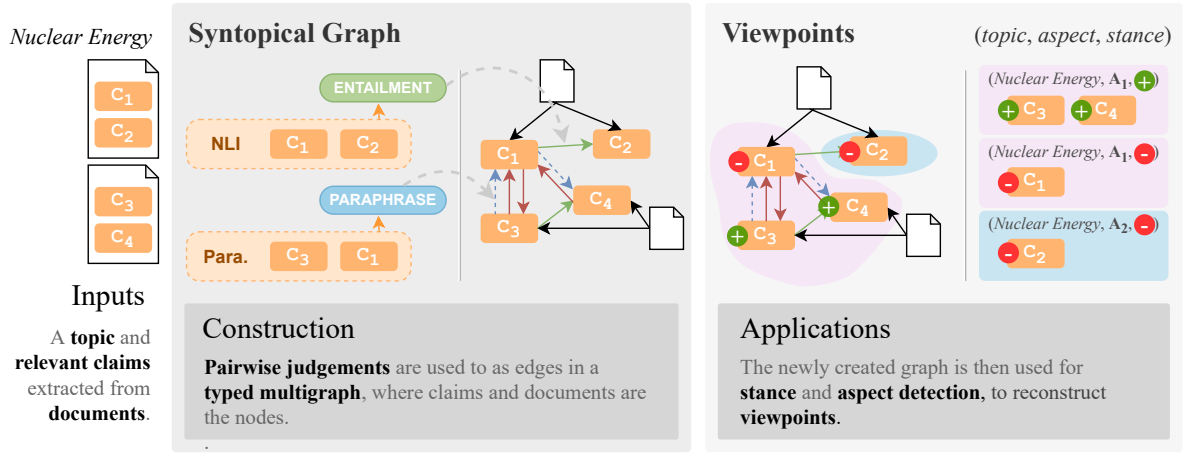

Figure 1: We introduce the idea of a *syntopical graph*, a data structure that represents the context of claims. The graph is a typed multi-graph (multiple edges allowed between nodes), where nodes are claims or documents, and edges are pairwise relationships such as entailment, paraphrase, topical similarity, or term similarity. By using this graph as input to graph neural networks or traditional graph algorithms, we can significantly improve on the tasks of aspect and stance detection, which allow us to identify viewpoints in a collection.

points underlying the claims in a collection, such that a reader can investigate and engage with them.

Many existing approaches attempt to identify viewpoints within a collection largely from the text of individual claims only, which we refer to as "content-only approaches." However, as the latent viewpoints are a global property of a collection, it is necessary to account not only for the text but also its context. For instance, in order to identify the stance of a claim with respect to a topic, it may help to consider the claim's stance relative to other claims on the topic. Although a few researchers have accounted for connections between claims and other information (details in Section 2), no systematic model of their interactions exists yet.

We therefore introduce a **syntopical graph** that models pairwise textual relationships between claims in order to enable a better reconstruction of the latent viewpoints in a collection. In line with the idea of Adler and Van Doren (1940), the syntopical graph makes the points of agreement and disagreement within the collection explicit. Technically, it denotes a multi-graph (where a pair of nodes can have many typed edges) that simultaneously represents relationships such as relative stance, relative specificity, or whether a claim paraphrases another. We build syntopical graphs by transferring pretrained pairwise models, requiring no additional training data to be annotated.

We decompose the problem of viewpoint reconstruction into the subtasks of *stance detection* and *aspect detection*, and evaluate the benefits of syntopical graphs — which are a collection-level approach — on both tasks. For stance detection, we use the sentential argumentation mining collection (Stab et al., 2018) and the IBM claim stance dataset (Bar-Haim et al., 2017a). For aspect detection we use the argument frames collection (Ajjour et al., 2019). We treat the graph as an input to: (a) a graph neural network architecture for stance detection, and (b) graph algorithms for unsupervised tasks such as aspect clustering. In both settings, our results show that the syntopical graph approach improves significantly over content-only baselines.

The contributions of the work are two-fold:

1. A well-motivated data structure for capturing the latent structure of an argumentative corpus, the syntopical graph.
2. An instantiation of syntopical graphs that yields state-of-the-art results on stance detection and aspect detection.

2 Related Work

First attempts at stance detection used content-oriented features (Somasundaran and Wiebe, 2009). Later approaches, such as those by Ranade et al. (2013) and Hasan and Ng (2013), exploited common patterns in dialogic structure to improve stance detection. More tailored to argumentation, Bar-Haim et al. (2017a) first identified the aspects of a discussed topic in two related claims and the sentiment towards these aspects. From this information, they derived stance based on the contrastiveness of the aspects. Later, Bar-Haim et al. (2017b) mod-

eled the context of a claim to account for cases without sentiment. Our work follows up on and generalizes this idea, systematically incorporating implicit and explicit structure induced by the topics, aspects, claims, and participants in a debate.

In a similar vein, Li et al. (2018) embedded debate posts and authors jointly based on their interactions, in order to classify a post's stance towards the debate topic. Durmus et al. (2019) encoded related pairs of claims using BERT to predict the stance and specificity of any claim in a complex structure of online debates. However, neither of these exploited the full graph structure resulting from all the relations and interactions in a debate, which is the gap we fill in this paper. Sridhar et al. (2015) model collective information about debate posts, authors, and their agreement and disagreement using probabilistic soft logic. Whereas they are restricted to the structure available in a forum, our approach can in principle be applied to arbitrary collections of text.

We also tackle aspect detection, which may at first seem more content-oriented in nature. Accordingly, previous research such as the works of Misra et al. (2015) and Reimers et al. (2019b) employed word-based features or contextualized word embeddings for topic-specific aspect clustering. Ajjour et al. (2019), whose *argument frames* dataset we use, instead clustered aspects with Latent Semantic Analysis (LSA) and topic modeling. But, in general, aspects might not be mentioned in a text explicitly. Therefore, we follow these other approaches, treating the task as a clustering problem. Unlike them, however, we do not model only the content and linguistic structure of texts, but we combine them with the debate structure.

Different types of argumentation graphs have been proposed, covering expert-stance information (Toledo-Ronen et al., 2016), basic argument and debate structure (Peldszus and Stede, 2015; Gemechu and Reed, 2019), specific effect relations (Al-Khatib et al., 2020; Kobbe et al., 2020), social media graphs (Aldayel and Magdy, 2019), and knowledge graphs (Zhang et al., 2020). Our main focus is not learning to construct ground-truth graphs, but how to use an approximated graph to derive properties such as stance and aspect. Our work resembles approaches that derive the relevance of arguments (Wachsmuth et al., 2017b) or their centrality and divisiveness in a discussion (Lawrence and Reed, 2017) from respective graphs. Sawhney et al. (2020) used a neural graph attention network to classify speech stance based on a graph with texts, speakers, and topics as nodes. While we also use a relational graph convolutional network for learning, the graph we propose captures implicit claim relations *as well as* explicit structure.

In addition, text-based graph neural models have been proposed to facilitate classification, such as TextGCN (Yao et al., 2019) as well as the follow-up work BertGCN (Lin et al., 2021). These approaches build a graph over terms (using normalized mutual information for edge weights) as well as sentences and documents (using TF-IDF for edge weights) to improve sentence- or document-level classification. Our work generalizes this approach, focusing on incorporating many edge types with different meanings, such as relative stance or relative specificity. We compare our approach with a BertGCN baseline, and we ablate all considered edge types, in order to show the importance of capturing these different textual relationships.

Ultimately, we seek to facilitate understanding of the main viewpoints in a text collection. Qiu and Jiang (2013) used clustering-based viewpoint discovery to study the impact of the interaction of topics and users in forum discussions. Egan et al. (2016) used multi-document summarization techniques to mine and organize the main points in a debate, and Vilares and He (2017) mined the main topics and their aspects using a Bayesian model. Bar-Haim et al. (2020) introduced the idea of keypoint analysis, grouping arguments found in a collection by the viewpoint they reflect and summarizing each group to a salient keypoint. While our graph-based analysis is likely to be suitable for finding keypoints, we instead focus on reconstructing latent viewpoints by grouping claims, leaving open the option to identify the key claims in future work as it would require manual evaluation.

3 Syntopical Graphs

We now introduce the concept of a *syntopical graph*. The goal of our syntopical graph is to systematically model the salient interactions of all claims in a collection of documents. Then, properties of claims (say, their stance towards a topic or the aspects they cover) can be assessed based not only on the content of the claim alone, but on the entirety of information available in their context.

To capture this context, we build a graph where documents and claims are nodes. Edges between

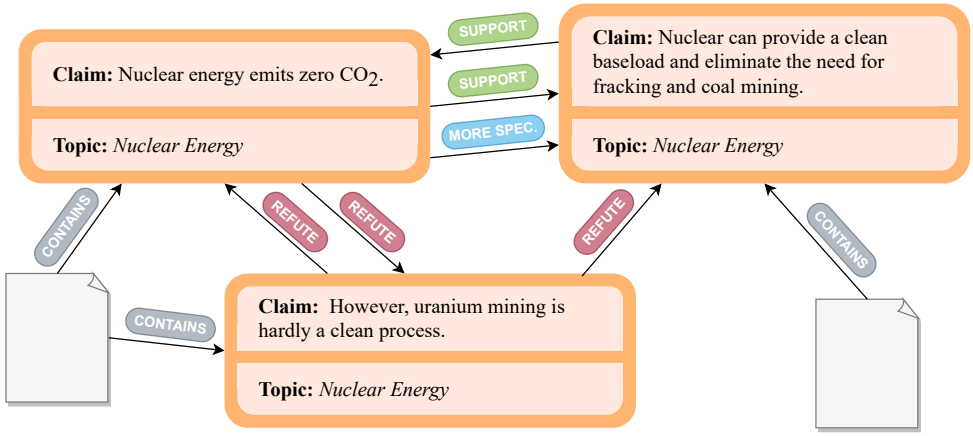

Figure 2: An example syntopical graph created from a collection of documents on the topic of *Nuclear Energy*. The nodes are documents and claims, and there are 0+ weighted and typed edges between any pair of nodes. In downstream applications, we add the representation of the topic to the claim nodes.

claims are constructed using pairwise scoring functions, such as pretrained natural language inference (NLI) models. Claims may relate to each other in many different ways: they can support or refute each other, they can paraphrase each other, they can entail or contradict each other, they can be topically similar, etc. We hypothesize that being able to account for these relationships helps computational argumentation tasks such as stance detection.

3.1 Graph Components

Intuitively, if it is known that claim (a) refutes claim (b), and claim (b) has a positive stance to the topic, it seems more reasonable to believe that claim (a) has a negative stance. We can represent all of this with a graph if we allow multiple edges between nodes. For instance, claims can have edges that label both relative agreement and relative specificity, as exemplified in the graph in Figure 2. The process of constructing a graph is shown in Figure 1.

Technically, we capture this intuition as a typed multi-graph: *typed* in that the nodes have different types drawn from $\{document, claim\}$, and a *multi-graph* because multiple edges (of different types) are allowed between nodes. We then formally define a syntopical graph as a labeled multi-graph in terms of a 5-tuple G:

$$G = (\Sigma_N, \Sigma_E, N, E, l_N, l_E),$$

where Σ_N is the alphabet of node types, Σ_E is the alphabet of edge types, N is the set of nodes, E is the set of multi-edges, $l_N : N \rightarrow \Sigma_N$ maps each node to its type, and $l_E : E \rightarrow \Sigma_E$ maps each edge to its type. In the following, we show how to construct the graph and what each of its components look like.

The node types, Σ_N, are used to represent structured metadata in the graph:

$$\Sigma_N = \{\text{claim}, \text{document}\}$$

Each node in the graph is mapped to its type with the function l_N. Accordingly, the edge alphabet is

$$\Sigma_E = \Sigma_{E:claim} \cup \Sigma_{E:document},$$

where $\Sigma_{E:claim}$ is the set of types of claim-claim edges and $\Sigma_{E:document}$ is the set of types of claim-document edges.

Claim Nodes The central node type in a syntopical graph is a claim node. A claim node represents a topically relevant claim in a collection. By treating a claim as a node embedded in a graph, we can take advantage of rich graph structures to represent the context in which the claim occurs, such as the document the claim appears in or the claim's relationship with other claims.

Document Nodes In general, two claims from the same source are more likely to represent the same viewpoint than a pair of claims sampled randomly. To capture this intuition, we allow claims from the same source to share information with each other via document nodes, which enables models to pool information about groups of claims and share the information amongst them. Similar information about claims can be aggregated in the metadata node and broadcast out to all claims.

Pairwise Relationships as Multi-Edges There are two classes of edge types:

- *claim-claim* edges ($\Sigma_{E:claim}$) model the relationship between pairs of claims: do they

support each other, is one more specific than the other, etc. Different tasks can make use of this information (e.g., a claim is likely to have a specific stance if other claims that support it have the same stance).

- *claim-document* edges ($\Sigma_{E:document}$) allow groups of claims to share information with each other through common ancestors (e.g., claims in a document pro nuclear energy are somewhat likely to have a pro stance).

Any pair of nodes can have multiple edges of different types between them; a claim can both contradict and refute another claim, for instance.

Edge Weights An edge can have a real-valued weight associated with it on the range $(-1, 1)$, representing the strength of the connection. The relative stance edge between a claim which strongly refutes another would receive a weight close to -1.

3.2 Graph Construction

For graph edges, we combine four pretrained models and two similarity measures. The pretrained edge types are: *relative stance* and *relative specificity* from Durmus et al. (2019), *paraphrase* edges from Dolan et al. (2004); Morris et al. (2020), and *natural language inference* edges from Williams et al. (2018); Liu et al. (2019). The edge weights are the confidence scores defined by

$$weight(u, v, r) = p_{pos(u,v)} - p_{neg(u,v)},$$

where u and v are claims, r is the relation type, and $p_{pos(u,v)}$ is the probability of a positive association between the claims (e.g., "is a paraphrase" or "does entail"), $p_{neg(u,v)}$ for a negative one. For similarity-based edges, we use standard TF-IDF for *term-based similarity* and LDA for *topic-based similarity* (Blei et al., 2003), using cosine similarity as the edge weight. The document-claim edges have a single type, *contains*, with an edge weight of 1. We compute each of the pairwise relationships for all pairs of claims that share the same topic, and then filter out edges using a threshold τ on the absolute value of the edge weight. τ is tuned as a hyperparameter on a validation dataset for each task.

For node representations, we initialize the claim node representations with the output of a natural language inference model that predicts whether the claim entails the topic. We initialize the document representations with a sentence vectorizer over the text of the document.

4 Viewpoint Reconstruction

A *viewpoint* can be understood as a judgment of some aspect of a topic that conveys a stance towards the topic. The goal of viewpoint reconstruction is to identify the set of viewpoints in a collection given a topic, starting with the claims. An example of this process is shown on the right in Figure 1. To denote viewpoints, we borrow notation in line with the idea of aspect-based argument mining (Trautmann, 2020), which in turn was inspired by aspect-based sentiment analysis. In particular, we express a viewpoint as a triple V:

$$V = (\text{topic}, \text{aspect}, \text{stance})$$

A *claim* is an expression of a viewpoint in natural language, and a single viewpoint can be expressed in several ways throughout a collection in many claims. *Aspects* are facets of the broader argument around the topic. While some actual claims may encode multiple viewpoints simultaneously, henceforth we consider each claim to encode one viewpoint for simplicity. To tackle viewpoint reconstruction computationally, we decompose it into two sub-tasks, stance detection and aspect detection, along with a final grouping of claims with same aspect and stance.

Stance Detection Stance detection requires assigning a valence label to a claim with respect to a particular topic. Though content-only baselines can work in many cases, there are also cases where the stance of a claim might only make sense in relation to a broader argument. For example, the claim "Nuclear power plants take 5 years to construct" is difficult to assign a stance a priori. However, in the context of other claims such as "Solar farms often take less than 2 years to commission", it might be viewed as having a negative stance. To exploit this additional contextual information, we use syntopical graphs as input to a graph neural network, in particular a Relational Graph Convolutional Network (R-GCN) (Schlichtkrull et al., 2018).

We treat stance detection as a supervised node classification task. The goal is to output a prediction in the set {PRO, CON} for each claim node relative to a topic. R-GCNs were developed to perform node classification and edge prediction for knowledge bases, which are also typed multigraphs. As such, the abstractions of the syntopical graph slot neatly into the abstractions of R-GCNs.

The input to an R-GCN is a weighted, typed multigraph with some initial node representation.

The network is made up of stacked relational graph convolutional layers; each layer computes a new set of node representations based on each node's neighborhood. In effect, each layer combines the edge-type-specific representation of all of a node's neighbors with its own representation. The representations are influenced by the node, and all of its neighbors, attenuated through the edge weight. An R-GCN thus consumes a set of initial claim representations, transforms them through stacks of relational graph convolutional layers, and outputs a final set of node vectors, which are fed into a classifier to predict the claim stance.

Aspect Detection Following the work of Ajjour et al. (2019), we treat aspect detection as an unsupervised task. As aspects are an open class, we use a community detection approach, modularity-based community detection (Clauset et al., 2004). The key intuition of modularity-based community detection is that communities are graph partitions that have more edges *within* communities than *across* communities. Modularity is a value assigned to a graph partition, which is higher when there are fewer edges across communities than within them; a modularity of 0 represents a random partition, while higher modularities indicate tighter communities. The goal of modularity-based community detection is to maximize modularity by finding dense partitions. This intuition works well for aspects in a syntopical graph — claims that discuss a similar aspect are likely to have salient interactions.

As aspects themselves are independent of stance, the direction of the interactions (e.g., support or refute) does not matter, but their salience does. To capture only the intensity of the interaction between two claims, we apply a transformation to signed collapse the multi-edges of a syntopical graph (denoted SG) to a positive-weighted graph (G):

$$w_G(u,v) = \frac{\sum_{t \in \Sigma_E} \delta_{SG}(u,v,t) \cdot |w_{SG}(u,v,t)|}{\sum_{t \in \Sigma_E} \delta_{SG}(u,v,t)},$$

where $w_G(u,v)$ is the weight between nodes u and v in the new graph G, $\delta_{SG}(u,v,t) = 1$ if an edge of type t exists between nodes u and v in the syntopical graph (SG), and $w_{SG}(u,v,t)$ is the edge weight for type t between nodes u and v in the syntopical graph. This is equivalent to taking the average across types of the absolute values of the weights. The newly constructed single-edge graph is then used to identify aspects, which should have more interactions between them than across them.

5 Experiments

To evaluate the effectiveness of our approach at reconstructing viewpoints, we consider three datasets across the two subtasks of stance and aspect detection. We hypothesize that syntopical graph approaches will outperform content-only baselines — including the ones used to initialize the claim representations — because they are able to make use of not only the claim content, but also the claim context. We further hypothesize that syntopical graph approaches will outperform graph-based baselines that use only textual similarity edges, because the latter's claim context is not as rich. For our experiments, we construct a syntopical graph as described in Section 3.

We further evaluate our model by conducting several additional experiments, including removing the use of document nodes or initial claim representations, analyzing the performance of each edge type in isolation and when left out, and an analysis of the differences in predictions between the syntopical graph and the content-only baselines.

Stance Detection For the stance detection experiments, we use two datasets: first, the heterogeneous cross-topic argumentation mining dataset (ArgMin) from Stab et al. (2018), and second, the claim-stance dataset (IBMCS) from Bar-Haim et al. (2017a). The ArgMin dataset contains about 25k sentences from 400 documents across eight controversial topics, ranging from abortion to school uniforms. Following Schiller et al. (2020), we filter only the claims, resulting in 11.1k claims. The IBMCS dataset contains 2.4k claims across 55 topics. We use the splits from Schiller et al. (2020), which ensure that the topics in the training and test sets are mutually exclusive. Claims are given a stance label drawn from {PRO, CON}. We evaluate using macro-averaged F_1 and accuracy.

We use a syntopical graph for each dataset as the input to a relational graph convolutional network (R-GCN), implemented in DGL (Wang et al., 2019) and PyTorch (Paszke et al., 2019). For document node representations, we use a pretrained sentence transformer and concatenate all of the sentences as input (Reimers et al., 2019a). For the claim node representations, we use a RoBERTa model pretrained on an NLI task (Liu et al., 2019) to encode both the claim and topic; the resulting vectors are fixed throughout training.

Model	IBMCS		ArgMin	
	macro F_1	Acc	macro F_1	Acc
Majority Baseline	34.06	51.66	33.83	51.14
RoBERTa Large NLI	52.34	52.69	60.56	60.93
BertGCN (Lin et al., 2021)	66.16	66.26	58.51	58.73
MT-DNN, 1 Dataset (Schiller et al., 2020)*	70.66	71.16	61.65	62.40
MT-DNN, 10 Datasets (Schiller et al., 2020)*	77.72	77.87	61.38	62.11
Syntopical Graph (R-GCN, Structure Only)	44.32	47.82	42.59	52.71
Syntopical Graph (R-GCN, No Documents)	83.03	83.10	67.52	**68.34**
Syntopical Graph (R-GCN)	**83.40**	**83.54**	**67.77**	68.01

Table 1: Results on the two **stance detection** datasets. The full syntopical graph, as well as the variant without document nodes, outperforms the content only baselines by both a significant and substantial margin ($p < 10^{-7}$ for ArgMin, and $p < 10^{-4}$ for IBMCS). A * on the model means we retrained a previously reported baseline.

Model	b-cubed F_1	b-cubed P	b-cubed R
LDA	47.01	47.19	49.82
Clustering (RoBERTa Large MNLI)	45.69	44.76	50.15
Syntopical Graph (Modularity)	**55.42**	**66.11**	**53.82**

Table 2: **Aspect detection** results on the argument frames dataset (Ajjour et al., 2019). The syntopical graph outperformed both LDA and clustering of RoBERTa embeddings, recovering latent aspects substantially better than either approach. The syntopical graph approach significantly outperforms LDA ($p < 10^{-19}$).

Aspect Detection For clustering-based aspect detection, we use the argument frames dataset from Ajjour et al. (2019). The dataset contains roughly 11k sentences drawn from 465 different topics. Each sentence has a specific aspect (or frame, in the original paper), drawn from a set of over a thousand possible aspects. Following the authors, we evaluate with a clustering metric, b-cubed F_1 (Amigó et al., 2009). We transform the graph as described in Section 4 to use as an input to modularity-based community detection, using τ of 0.6 tuned on held-out topics.

6 Results and Analysis

The main results for stance detection are shown in Table 1. The most important finding is that the fusion of signals from content and from structure done by our approach *syntopical graph (R-GCN)* outperforms the existing state-of-the-art (Schiller et al., 2020) for both the IBMCS dataset (83.40 macro F_1, +5.68 absolute) and the ArgMin dataset (67.7 macro F_1, +6.12 absolute). The content-oriented *RoBERTa Large NLI* model and the structure-only syntopical graph have significantly reduced performance independently, emphasizing the complementarity of the two signals. Our best network is the one which includes both claim and document node, except for the ArgMin dataset.

Aspect detection results are shown in Table 2. Our modularity approach outperforms the state-of-the-art (Ajjour et al., 2019) on the argument frames dataset (55.42 b-cubed F_1, +8.41 absolute).

The remainder of this section investigates the robustness of the syntopical graph approach to stance and aspect detection: First, we analyze the contribution of each edge type, running experiments without and with only each edge type. We also examine the accuracy of the edges in our graph when applied out of domain as well as analysis to understand the types of claims for which this model improves performance.

Edge Analysis We conducted an ablation study to analyze the usefulness of each considered edge type. To do so, we built graphs containing each edge independently, and graphs dropping each edge independently. Table 3 presents the results.

For the supervised task of stance detection, we use the IBMCS dataset. No single edge performs as well as the combination of edges, the best being *relative stance* with a macro-F_1 score of 80.72. This indicates that our model is capable of taking advantage of the different kinds of relationships represented by the edge types. We see the largest performance drops when we remove *relative stance* (79.39), *relative specificity* (79.39), or *NLI* (78.95) edges respectively, indicating the highest amount of unique information being captured by these edges. In contrast, *paraphrase* can be removed without loss for stance detection according to the results.

Edge	Model	Stance Detection (macro F_1)		Aspect Detection	
		Alone	Without	Alone	Without
Relative Stance	RoBERTa Base	80.72	79.39	52.22	53.52
Relative Specificity	RoBERTa Base	70.22	79.35	43.59	55.73
Paraphrase	RoBERTa Large	75.57	83.42	56.31	53.77
NLI	RoBERTa Base	80.29	78.95	53.16	53.52
Term Similarity	TF-IDF + *cosine*	73.62	81.83	52.40	54.74
Topic Similarity	LDA + *cosine*	72.67	82.54	51.11	54.92
All Edges		**83.40**		**55.42**	

Table 3: Importance of each edge type for both evaluated tasks. We examine each edge type *alone* and when eliminated from the graph entirely (*without*). For supervised stance detection, no single edge performs as well as the combination of all edges. For unsupervised aspect detection paraphrase edges provide the best signal.

Edge (All RoBERTa)	Accuracy		
	Top	Bottom	Random
Stance	53%	44%	52%
Specificity	82%	52%	56%
Paraphrase	93%	65%	74%
NLI	93%	50%	60%

Table 4: Performance of each edge type across domains considering the 100 strongest edges, the 100 weakest edges, and 100 random edges. There is a clear trend of the strongest edges being more accurate and the weakest edges being less accurate, meaning that the edge weight does have some predictive effect about the edge's accuracy.

This is opposite for aspect detection, which we treat as an unsupervised community detection task; here *paraphrase* alone outperforms the graph with all edge relationships (macro F_1 56.31 versus 55.42). The other edges even have a slight negative effect on the overall results (55.42); being unsupervised, our approach here has no way of filtering out uninformative edges.

Edge Domain Transfer One possible confounder of the contribution of each edge type is the out-of-domain performance of the pairwise model used to predict that edge. A poor model would provide little more than random noise, even if the edge type were expected to be helpful. To investigate this possibility, we sampled 100 each of the edges (above $\tau = 0.6$) with the highest weight, the lowest weight, and a random sample. We then annotated each edge as being correctly or incorrectly predicted. Results are shown in Table 4.

There is a clear trend that the edge weight is correlated with edge correctness, meaning that the models retain some level of calibration across domains. As we incorporate the edge weight in the R-GCN, this helps to lessen the effect of the noisier, weaker edges. Another trend is that an edge type's usefulness across tasks is not solely a function of that edge type's accuracy. The type of failure mode is also important. For instance, the relative stance edges have poor surface-level accuracy, but the most common failure was not predicting the wrong relative stance; it was predicting *any* stance for pairs of claims about different aspects.

Flip Analysis Finally, we analyze "flipped" cases in stance detection in which the baseline predicted stance incorrectly but the model predicted stance correctly, or vice-versa, to understand areas for which this model improves performance. A sample of these is shown in Table 5.

Perhaps the most surprising result is how *different* the predictions of the syntopical graph-based approach are from those of the content-only MT-DNN baseline. For the IBMCS dataset, there were 1355 claims in the test set, and we flipped 219 (16.2%) correctly relative to the MT-DNN baseline, but also 140 (10.3%) *incorrectly* compared to that baseline. Thus, we flipped 26.5% of the overall predictions for the 5.68 point improvement in F_1. This holds across the ArgMin dataset as well, where we flipped 536 (19.6%) claims correctly and 373 (13.7%) claims incorrectly, out of a total 2726 claims in the test set. Though we show substantial gains overall, it seems that the models capture different signals. We thus believe that future improvements through improved model combination may still be possible.

7 Conclusion

In this paper, we have introduced a data structure, the *syntopical graph*, which provides context for claims in collections. We have provided empirical evidence that syntopical graphs can be used as input representations for graph-structured approaches

Example	True	MT-DNN	Syn. Gr.	Reason
Topic: wind power should be a primary focus of future energy supply *Claim*: predictability of wind plant output remains low *Strongest Neighbor (+)*: the non-dispatchable nature of wind energy production can raise costs	CON	PRO	CON	Good Neighbors
Topic: wind power should be a primary focus of future energy supply *Claim*: Wind power uses little land *Strongest Neighbor (-)*: wind power "cannot be relied upon to provide significant levels of power	PRO	PRO	CON	Bad Neighbors
Topic: build the Keystone XL pipeline *Claim*: the pipeline would be "game over for the planet *Strongest Neighbor (-)*: this is the most technologically advanced and safest pipeline ever proposed	CON	PRO	CON	Good Neighbors

Table 5: Stance detection examples with the *true* stance label where the output label of our *syntopical graph* was different from that of the *MT-DNN* baseline, along with a potential *reason*.

(such as graph neural networks and graph clustering algorithms) to obtain significant improvements over content-only baselines.

We believe there are several opportunities to extend this work in the future. First, we believe the graph construction could be improved by avoiding the inefficient pairwise analysis, expanding the edge types, and utilizing a more robust classifier for the graph. Second, we would relax the constraint that a claim represents a single viewpoint, or the limitation of aspect detection to unsupervised approaches. Finally, we would like to apply our approach to the original problem first motivated by syntopical reading to see if this system can aid users in browsing or understanding a collection.

8 Ethics Impact Statement

We anticipate that the syntopical graph explored in this work will have a beneficial impact in real world systems to aid users in improved comprehension and reduce susceptibility to misinformation. The goal of our work is motivated by syntopical reading, which theorizes that individuals exposed to agreement and disagreement within a collection gain a deeper understanding of the central topics. Our work on syntopical graphs provides an algorithmic foundation to aid readers in understanding the key viewpoints (aspect and stance for a given topic) present in a collection.

Acknowledgments

We would like to thank many others for their invaluable feedback and patient discussions, including Charlotte Ellison, Ani Nenkova, Tong Sun, Han-Chin Shing, and Pedro Rodriguez. This work was generously supported through Adobe Gift Funding, which supports an Adobe Research-University of Maryland collaboration. It was completed while the primary author was interning at Adobe Research.

References

Mortimer J Adler and Charles Van Doren. 1940. *How To Read A Book*. Simon and Schuster, New York.

Yamen Ajjour, Milad Alshomary, Henning Wachsmuth, and Benno Stein. 2019. Modeling frames in argumentation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2922–2932, Hong Kong, China. Association for Computational Linguistics.

Khalid Al-Khatib, Yufang Hou, Henning Wachsmuth, Charles Jochim, Francesca Bonin, and Benno Stein. 2020. End-to-end argumentation knowledge graph construction. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 7367–7374. AAAI.

Abeer Aldayel and Walid Magdy. 2019. Your stance is exposed! analysing possible factors for stance detection on social media. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–20.

Enrique Amigó, Julio Gonzalo, Javier Artiles, and Felisa Verdejo. 2009. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information retrieval*, 12(4):461–486.

Roy Bar-Haim, Indrajit Bhattacharya, Francesco Dinuzzo, Amrita Saha, and Noam Slonim. 2017a. Stance classification of context-dependent claims. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 251–261, Valencia, Spain. Association for Computational Linguistics.

Roy Bar-Haim, Lilach Edelstein, Charles Jochim, and Noam Slonim. 2017b. Improving claim stance classification with lexical knowledge expansion and context utilization. In *Proceedings of the 4th Workshop on Argument Mining*, pages 32–38, Copenhagen, Denmark. Association for Computational Linguistics.

Roy Bar-Haim, Yoav Kantor, Lilach Eden, Roni Friedman, Dan Lahav, and Noam Slonim. 2020. Quantitative argument summarization and beyond: Cross-domain key point analysis. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 39–49, Online. Association for Computational Linguistics.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Sihao Chen, Daniel Khashabi, Chris Callison-Burch, and Dan Roth. 2019. PerspectroScope: A window to the world of diverse perspectives. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 129–134, Florence, Italy. Association for Computational Linguistics.

Aaron Clauset, Mark EJ Newman, and Cristopher Moore. 2004. Finding community structure in very large networks. *Physical review E*, 70(6):066111.

Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 350–356, Geneva, Switzerland. COLING.

Esin Durmus, Faisal Ladhak, and Claire Cardie. 2019. Determining relative argument specificity and stance for complex argumentative structures. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4630–4641, Florence, Italy. Association for Computational Linguistics.

Charlie Egan, Advaith Siddharthan, and Adam Wyner. 2016. Summarising the points made in online political debates. In *Proceedings of the Third Workshop on Argument Mining (ArgMining2016)*, pages 134–143, Berlin, Germany. Association for Computational Linguistics.

Debela Gemechu and Chris Reed. 2019. Decompositional argument mining: A general purpose approach for argument graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 516–526, Florence, Italy. Association for Computational Linguistics.

Kazi Saidul Hasan and Vincent Ng. 2013. Stance classification of ideological debates: Data, models, features, and constraints. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1348–1356, Nagoya, Japan. Asian Federation of Natural Language Processing.

Jonathan Kobbe, Ioana Hulpuș, and Heiner Stuckenschmidt. 2020. Unsupervised stance detection for arguments from consequences. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 50–60.

John Lawrence and Chris Reed. 2017. Using complex argumentative interactions to reconstruct the argumentative structure of large-scale debates. In *Proceedings of the 4th Workshop on Argument Mining*, pages 108–117, Copenhagen, Denmark. Association for Computational Linguistics.

Chang Li, Aldo Porco, and Dan Goldwasser. 2018. Structured representation learning for online debate stance prediction. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3728–3739, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Yuxiao Lin, Yuxian Meng, Xiaofei Sun, Qinghong Han, Kun Kuang, Jiwei Li, and Fei Wu. 2021. BertGCN: Transductive text classification by combining GCN and BERT. *arXiv preprint arXiv:2105.05727*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Amita Misra, Pranav Anand, Jean E. Fox Tree, and Marilyn Walker. 2015. 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. Association for Computational Linguistics.

John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Andreas Peldszus and Manfred Stede. 2015. Joint prediction in MST-style discourse parsing for argumentation mining. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language*

Processing, pages 938–948, Lisbon, Portugal. Association for Computational Linguistics.

Minghui Qiu and Jing Jiang. 2013. A latent variable model for viewpoint discovery from threaded forum posts. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1031–1040, Atlanta, Georgia. Association for Computational Linguistics.

Sarvesh Ranade, Rajeev Sangal, and Radhika Mamidi. 2013. Stance classification in online debates by recognizing users' intentions. In *Proceedings of the SIGDIAL 2013 Conference*, pages 61–69, Metz, France. Association for Computational Linguistics.

Nils Reimers, Iryna Gurevych, Nils Reimers, Iryna Gurevych, Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. 2019a. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Nils Reimers, Benjamin Schiller, Tilman Beck, Johannes Daxenberger, Christian Stab, and Iryna Gurevych. 2019b. Classification and clustering of arguments with contextualized word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 567–578, Florence, Italy. Association for Computational Linguistics.

Ramit Sawhney, Arnav Wadhwa, Shivam Agarwal, and Rajiv Ratn Shah. 2020. GPolS: A contextual graph-based language model for analyzing parliamentary debates and political cohesion. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4847–4859, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Benjamin Schiller, Johannes Daxenberger, and Iryna Gurevych. 2020. Stance detection benchmark: How robust is your stance detection? *CoRR*, abs/2001.01565.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer.

Swapna Somasundaran and Janyce Wiebe. 2009. 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. Association for Computational Linguistics.

Dhanya Sridhar, James Foulds, Bert Huang, Lise Getoor, and Marilyn Walker. 2015. Joint models of disagreement and stance in online debate. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 116–125, Beijing, China. Association for Computational Linguistics.

Christian Stab, Tristan Miller, Benjamin Schiller, Pranav Rai, and Iryna Gurevych. 2018. Cross-topic argument mining from heterogeneous sources. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3664–3674, Brussels, Belgium. Association for Computational Linguistics.

Manfred Stede and Jodi Schneider. 2018. *Argumentation Mining*. Number 40 in Synthesis Lectures on Human Language Technologies. Morgan & Claypool.

Orith Toledo-Ronen, Roy Bar-Haim, and Noam Slonim. 2016. Expert stance graphs for computational argumentation. In *Proceedings of the Third Workshop on Argument Mining (ArgMining2016)*, pages 119–123, Berlin, Germany. Association for Computational Linguistics.

Dietrich Trautmann. 2020. Aspect-based argument mining. In *Proceedings of the 7th Workshop on Argument Mining*, pages 41–52, Online. Association for Computational Linguistics.

David Vilares and Yulan He. 2017. Detecting perspectives in political debates. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1573–1582, Copenhagen, Denmark. Association for Computational Linguistics.

Henning Wachsmuth, Martin Potthast, Khalid Al-Khatib, Yamen Ajjour, Jana Puschmann, Jiani Qu, Jonas Dorsch, Viorel Morari, Janek Bevendorff, and Benno Stein. 2017a. Building an argument search engine for the web. In *Proceedings of the 4th Workshop on Argument Mining*, pages 49–59. Association for Computational Linguistics.

Henning Wachsmuth, Benno Stein, and Yamen Ajjour. 2017b. "pageRank" for argument relevance. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1117–1127. Association for Computational Linguistics.

Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. 2019. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122. Association for Computational Linguistics.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7370–7377.

Bowen Zhang, Min Yang, Xutao Li, Yunming Ye, Xiaofei Xu, and Kuai Dai. 2020. Enhancing cross-target stance detection with transferable semantic-emotion knowledge. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3188–3197.

A Relational Graph Convolutional Networks

The input to an R-GCN is a weighted, typed multi-graph with some initial node representation. The network is made up of stacked relational graph convolutional layers; each layer computes a new set of node representations based on each node's neighborhood. In effect, each layer combines the edge-type-specific representation of all of a node's neighbors with its own representation. The propagation equation is defined per Schlichtkrull et al. (2018):

$$h_u^{(l+1)} = \sigma\left(\sum_{r\in\Sigma_E}\sum_{v\in\mathcal{N}_u^r}\frac{1}{|\mathcal{N}_u^r|}W_r^{(l)}h_v^{(l)}w_{u,v,r} + W_0^{(l)}h_u^{(l)}\right)$$

where u and v are nodes in the graph, $\mathcal{N}_u^r$ is the neighborhood for node u of edge types r, $\frac{1}{|\mathcal{N}_u^r|}$ is the normalization term, W_r is the per-relationship transformation, $w_{u,v,r}$ is the edge weight between nodes u and v of edge type r, and W_0 is the self-loop weight.

B Claim Node Representations

For the claim node representations, we format the input to the *Roberta Large NLI* model as:

```
[CLS] claim [SEP] topic [SEP]
```

We use the output representations (1024 dims per claim node) as the node representations for the graph.

C Hyperparameter Tuning

To tune hyperparameters, we used Optuna[1] and the tree of parzen estimators optimizer. We tuned the IBMCS dataset with 100 samples on a 1080Ti, training 10 epochs for each sample. For the ArgMin dataset, we tuned for 3 samples on an Nvidia Quadro RTX 6000, fixing all parameters from the best IBMCS dataset, except for the number of layers. We selected each based on the lowest validation loss.

D Selected Models

For both datasets, we tune the R-GCN on the validation set, ending up with the following parameter settings: number of 3 graph convolutional layers for ArgMin and 2 for IBMCS; 128 hidden dimensions per layer; a learning rate or 0.00856 and decay (γ) of 0.797; dropout of 0.005; τ of 0.6; batch size of 10; and 4 bases for edge relations. We trained each model for 10 epochs. The IBMCS model took roughly 20 minutes to train, and the ArgMin model took roughly 3 and a half hours to train. We ran each model 5 times to account for random variations, and selected the run with the lowest validation score.

The IBMCS model has roughly 248k parameters and the ArgMin model has roughly 330k tunable parameters.

The BertGCN baseline used the RoBERTaGCN configuration from Lin et al. (2021). Per the original paper, we first trained a RoBERTa model on the task for 50 epochs using a batch size of 64 and a learning rate of 0.00001, then trained the RoBERTaGCN model for 60 epochs using a batch size of 8, a GCN learning rate of 0.001, and a RoBERTa learning rate of 0.00001.

Parameter	Type	Low	High
Threshold (τ)	float	0.5	1.0
Learning Rate	float (log)	10^{-6}	10^{-2}
LR Decay (γ)	float	0.6	1.0
Hidden Layers	int	1	3
Hidden Units	int	50	200
Number of Bases	int	1	6
Dropout	float	0	0.5

Table 6: The range of hyperparameters we sweep over when training the relational graph convolutional network.

[1] https://optuna.org