Argument Mining with Structured SVMs and RNNs

Vlad Niculae
Cornell University
vlad@cs.cornell.edu

Joonsuk Park
Williams College
jpark@cs.williams.edu

Claire Cardie
Cornell University
cardie@cs.cornell.edu

Abstract

We propose a novel factor graph model for argument mining, designed for settings in which the argumentative relations in a document do not necessarily form a tree structure. (This is the case in over 20% of the web comments dataset we release.) Our model jointly learns elementary unit type classification and argumentative relation prediction. Moreover, our model supports SVM and RNN parametrizations, can enforce structure constraints (e.g., transitivity), and can express dependencies between adjacent relations and propositions. Our approaches outperform unstructured baselines in both web comments and argumentative essay datasets.

1 Introduction

Argument mining consists of the automatic identification of argumentative structures in documents, a valuable task with applications in policy making, summarization, and education, among others. The argument mining task includes the tightly-knit subproblems of classifying propositions into elementary unit types and detecting argumentative relations between the elementary units. The desired output is a document argumentation graph structure, such as the one in Figure 1, where propositions are denoted by letter subscripts, and the associated argumentation graph shows their types and support relations between them.

Most annotation and prediction efforts in argument mining have focused on tree or forest structures (Peldszus and Stede, 2015; Stab and Gurevych, 2016), constraining argument structures to form one or more trees. This makes the problem computationally easier by enabling the use of maximum spanning tree–style parsing approaches. However, argumentation *in the wild* can be less well-formed. The argument put forth in Figure 1, for instance, consists of two components: a simple tree structure and a more complex graph structure (c jointly supports b and d). In this work, we design a flexible and highly expressive structured prediction model for argument mining, jointly learning to classify elementary units (henceforth *propositions*) and to identify the argumentative relations between them (henceforth *links*). By formulating argument mining as inference in a factor graph (Kschischang et al., 2001), our model (described in Section 4) can account for correlations between the two tasks, can consider second order link structures (e.g., in Figure 1, $c \rightarrow b \rightarrow a$), and can impose arbitrary constraints (e.g., transitivity).

To parametrize our models, we evaluate two alternative directions: linear structured SVMs

[Calling a debtor at work is counter-intuitive;]$_a$ [if collectors are continuously calling someone at work, other employees may report it to the debtor's supervisor.]$_b$ [Most companies have established rules about receiving or making personal calls during working hours.]$_c$ [If a collector or creditor calls a debtor on his/her cell phone and is informed that the debtor is at work, the call should be terminated.]$_d$ [No calls to employers should be allowed,]$_e$ [as this jeopardizes the debtor's job.]$_f$

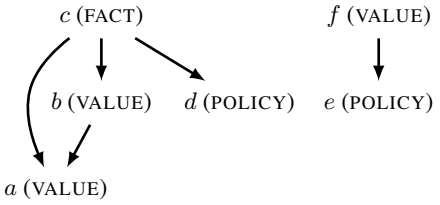


Figure 1: Example annotated CDCP comment.[1]

[1] We describe proposition types (FACT, etc.) in Section 3.

Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, pages 985–995
Vancouver, Canada, July 30 - August 4, 2017.
https://doi.org/10.18653/v1/P17-1091

(Tsochantaridis et al., 2005), and recurrent neural networks with structured loss, extending (Kiperwasser and Goldberg, 2016). Interestingly, RNNs perform poorly when trained with classification losses, but become competitive with the feature-engineered structured SVMs when trained within our proposed structured learning model.

We evaluate our approach on two argument mining datasets. Firstly, on our new *Cornell eRulemaking Corpus – CDCP*,[2] consisting of argument annotations on comments from an eRulemaking discussion forum, where links don't always form trees (Figure 1 shows an abridged example comment, and Section 3 describes the dataset in more detail). Secondly, on the UKP argumentative essays v2 (henceforth UKP), where argument graphs are annotated strictly as multiple trees (Stab and Gurevych, 2016). In both cases, the results presented in Section 5 confirm that our models outperform unstructured baselines. On UKP, we improve link prediction over the best reported result in (Stab and Gurevych, 2016), which is based on integer linear programming postprocessing. For insight into the strengths and weaknesses of the proposed models, as well as into the differences between SVM and RNN parameterizations, we perform an error analysis in Section 5.1. To support argument mining research, we also release our Python implementation, `Marseille`.[3]

2 Related work

Our factor graph formulation draws from ideas previously used independently in parsing and argument mining. In particular, maximum spanning tree (MST) methods for arc-factored dependency parsing have been successfully used by McDonald et al. (2005) and applied to argument mining with mixed results by Peldszus and Stede (2015). As they are not designed for the task, MST parsers cannot directly handle proposition classification or model the correlation between proposition and link prediction—a limitation our model addresses. Using RNN features in an MST parser with a structured loss was proposed by Kiperwasser and Goldberg (2016); their model can be seen as a particular case of our factor graph approach, limited to link prediction with a tree structure constraint. Our models support multi-task learning for proposition classification, parameterizing adjacent links with higher-order structures (e.g., $c \rightarrow b \rightarrow a$) and enforcing arbitrary constraints on the link structure, not limited to trees. Such higher-order structures and logic constraints have been successfully used for dependency and semantic parsing by Martins et al. (2013) and Martins and Almeida (2014); to our knowledge we are the first to apply them to argument mining, as well as the first to parametrize them with neural networks. Stab and Gurevych (2016) used an integer linear program to combine the output of independent proposition and link classifiers using a hand-crafted scoring formula, an approach similar to our baseline. Our factor graph method can combine the two tasks in a more principled way, as it fully learns the correlation between the two tasks without relying on hand-crafted scoring, and therefore can readily be applied to other argumentation datasets. Furthermore, our model can enforce the tree structure constraint, required on the UKP dataset, using MST cycle constraints used by Stab and Gurevych (2016), thanks to the AD^3 inference algorithm (Martins et al., 2015).

Sequence tagging has been applied to the related structured tasks of proposition identification and classification (Stab and Gurevych, 2016; Habernal and Gurevych, 2016; Park et al., 2015b); integrating such models is an important next step. Meanwhile, a new direction in argument mining explores *pointer networks* (Potash et al., 2016); a promising method, currently lacking support for tree structures and domain-specific constraints.

3 Data

We release a new argument mining dataset consisting of user comments about rule proposals regarding Consumer Debt Collection Practices (CDCP) by the Consumer Financial Protection Bureau collected from an eRulemaking website, `http://regulationroom.org`.

Argumentation structures found in web discussion forums, such as the eRulemaking one we use, can be more free-form than the ones encountered in controlled, elicited writing such as (Peldszus and Stede, 2015). For this reason, we adopt the model proposed by Park et al. (2015a), which does not constrain links to form tree structures, but unrestricted directed graphs. Indeed, over 20% of the comments in our dataset exhibit local structures that would not be allowable in a tree. Possible link types are *reason* and *evidence*, and propo-

[2] Dataset available at `http://joonsuk.org`.

[3] Available at `https://github.com/vene/marseille`.

sition types are split into five fine-grained categories: POLICY and VALUE contain subjective judgements/interpretations, where only the former specifies a specific course of action to be taken. On the other hand, TESTIMONY and FACT do not contain subjective expressions, the former being about personal experience, or "anecdotal." Lastly, REFERENCE covers URLs and citations, which are used to point to objective evidence in an online setting.

In comparison, the UKP dataset (Stab and Gurevych, 2016) only makes the syntactic distinction between CLAIM, MAJOR CLAIM, and PREMISE types, but it also includes *attack* links. The permissible link structure is stricter in UKP, with links constrained in annotation to form one or more disjoint directed trees within each paragraph. Also, since web arguments are not necessarily fully developed, our dataset has many argumentative propositions that are not in any argumentation relations. In fact, it isn't unusual for comments to have no argumentative links at all: 28% of CDCP comments have no links, unlike UKP, where all essays have complete argument structures. Such comments with no links make the problem harder, emphasizing the importance of capturing the *lack* of argumentative support, not only its presence.

3.1 Annotation results

Each user comment was annotated by two annotators, who independently annotated the boundaries and types of propositions, as well as the links among them.[4] To produce the final corpus, a third annotator manually resolved the conflicts,[5] and two automatic preprocessing steps were applied: we take the link transitive closure, and we remove a small number of nested propositions.[6] The resulting dataset contains 731 comments, consisting of about 3800 sentences (≈4700 propositions) and 88k words. Out of the 43k possible pairs of propositions, links are present between only 1300 (roughly 3%). In comparison, UKP has fewer documents (402), but they are longer, with a total of 7100 sentences (6100 propositions) and 147k words. Since UKP links only occur within the same paragraph and propositions not connected to the argument are removed in a preprocessing step, link prediction is less imbalanced in UKP, with 3800 pairs of propositions being linked out of a total of 22k (17%). We reserve a test set of 150 documents (973 propositions, 272 links) from CDCP, and use the provided 80-document test split from UKP (1266 propositions, 809 links).

[4]The annotators used the GATE annotation tool (Cunningham et al., 2011).

[5]Inter-annotator agreement is measured with Krippendorff's α (Krippendorff, 1980) with respect to elementary unit type (α=64.8%) and links (α=44.1%). A separate paper describing the dataset is under preparation.

[6]When two propositions overlap, we keep the one that results in losing the fewest links. For generality, we release the dataset without this preprocessing, and include code to reproduce it; we believe that handling nested argumentative units is an important direction for further research.

4 Structured learning for argument mining

4.1 Preliminaries

Binary and multi-class classification have been applied with some success to proposition and link prediction separately, but we seek a way to jointly learn the argument mining problem at the *document* level, to better model contextual dependencies and constraints. We therefore turn to *structured learning*, a framework that provides the desired level of expressivity.

In general, learning from a dataset of documents $x_i \in \mathcal{X}$ and their associated labels $y_i \in \mathcal{Y}$ involves seeking model parameters $\boldsymbol{w}$ that can "pick out" the best label under a scoring function f:

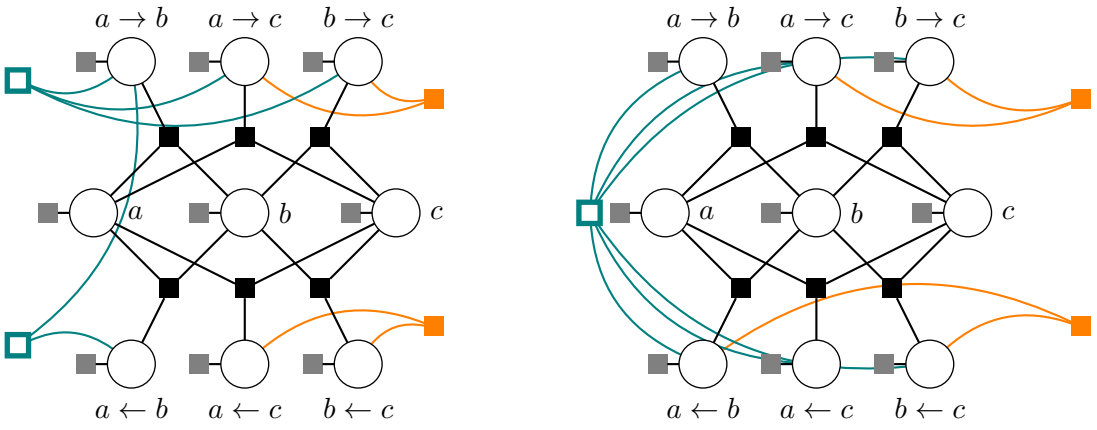
$$\hat{y} := \arg\max_{y \in \mathcal{Y}} f(x, y; \boldsymbol{w}). \quad (1)$$

Unlike classification or regression, where $\mathcal{X}$ is usually a feature space $\mathbb{R}^d$ and $\mathcal{Y} \subseteq \mathbb{R}$ (e.g., we predict an integer class index or a probability), in structured learning, more complex inputs and outputs are allowed. This makes the $\arg\max$ in Equation 1 impossible to evaluate by enumeration, so it is desirable to find models that decompose over smaller units and dependencies between them; for instance, as *factor graphs*. In this section, we give a factor graph description of our proposed structured model for argument mining.

4.2 Model description

An input document is a string of words with proposition offsets delimited. We denote the propositions in a document by $\{a, b, c, ...\}$ and the possible directed link between a and b as $a \to b$. The argument structure we seek to predict consists of the type of each proposition $y_a \in \mathcal{P}$ and a binary label for each link $y_{a \to b} \in \mathcal{R} = \{\texttt{on}, \texttt{off}\}$.[7]

[7]For simplicity and comparability, we follow Stab and Gurevych (2016) in using binary link labels even if links could be of different types. This can be addressed in our model by incorporating "labeled link" factors.



(a) CDCP (b) UKP

Figure 2: Factor graphs for a document with three propositions (a, b, c) and the six possible edges between them, and some of the factors used, illustrating differences and similarities between our models for the two datasets. Unary factors are light gray; compatibility factors are black. Factors not part of the basic model have curved edges: higher-order factors are orange and on the right; link structure factors are hollow, as that they don't have any parameters. Strict constraint factors are omitted for simplicity.

The possible proposition types $\mathcal{P}$ differ for the two datasets; such differences are documented in Table 1. As we describe the variables and factors constituting a document's factor graph, we shall refer to Figure 2 for illustration.

Unary potentials. Each proposition a and each link $a \to b$ has a corresponding random variable in the factor graph (the circles in Figure 2). To encode the model's belief in each possible value for these variables, we parametrize the *unary factors* (gray boxes in Figure 2) with unary potentials: $\boldsymbol{\phi}(a) \in \mathbb{R}^{|\mathcal{P}|}$ is a score of y_a for each possible proposition type. Similarly, link unary potentials $\boldsymbol{\phi}(a \to b) \in \mathbb{R}^{|\mathcal{R}|}$ are scores for $y_{a \to b}$ being `on`/`off`. Without any other factors, this would amount to independent classifiers for each task.

Compatibility factors. For every possible link $a \to b$, the variables $(a, b, a \to b)$ are bound by a dense factor scoring their joint assignment (the black boxes in Figure 2). Such a factor could automatically learn to encourage links from compatible types (e.g., from TESTIMONY to POLICY) or discourage links between less compatible ones (e.g., from FACT to TESTIMONY). In the simplest form, this factor would be parametrized as a tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{|\mathcal{P}| \times |\mathcal{P}| \times |\mathcal{R}|}$, with t_{ijk} retaining the score of a source proposition of type i to be (k = `on`) or not to be (k = `off`) in a link with a proposition of type j. For more flexibility, we parametrize this factor with **compatibility features** depending only on simple structure: $\boldsymbol{t}_{ijk}$ becomes a vector, and the score of configuration (i, j, k) is given by $\boldsymbol{v}_{ab}^{\top} \boldsymbol{t}_{ijk}$ where $\boldsymbol{v}_{ab}$ consists of three binary features:

- **bias:** a constant value of 1, allowing $\boldsymbol{\mathcal{T}}$ to learn a base score for a label configuration (i, j, k), as in the simple form above,
- **adjacency:** when there are no other propositions between the source and the target,
- **order:** when the source precedes the target.

Second order factors. Local argumentation graph structures such as $a \to b \to c$ might be modeled better together rather than through separate link factors for $a \to b$ and $b \to c$. As in higher-order structured models for semantic and dependency parsing (Martins et al., 2013; Martins and Almeida, 2014), we implement three types of second order factors: **grandparent** $(a \to b \to c)$, **sibling** $(a \leftarrow b \to c)$, and **co-parent** $(a \to b \leftarrow c)$. Not all of these types of factors make sense on all datasets: as sibling structures cannot exist in directed trees, we don't use sibling factors on UKP. On CDCP, by transitivity, every grandparent structure implies a corresponding sibling, so it is sufficient to parametrize siblings. This difference between datasets is emphasized in Figure 2, where one example of each type of factor is pictured on the right side of the graphs (orange boxes with curved edges): on CDCP we illustrate a co-parent factor (top right) and a sibling factor (bot-

tom right), while on UKP we show a co-parent factor (top right) and a grandparent factor (bottom right). We call these factors *second order* because they involve two link variables, scoring the joint assignment of both links being on.

Valid link structure. The global structure of argument links can be further constrained using domain knowledge. We implement this using constraint factors; these have no parameters and are denoted by empty boxes in Figure 2. In general, well-formed arguments should be cycle-free. In the UKP dataset, links form a directed forest and can never cross paragraphs. This particular constraint can be expressed as a series of *tree factors*,[8] one for each paragraph (the factor connected to all link variables in Figure 2). In CDCP, links do not form a tree, but we use logic constraints to enforce transitivity (top left factor in Figure 2) and to prevent symmetry (bottom left); the logic formulas implemented by these factors are described in Table 1. Together, the two constraints have the desirable side effect of preventing cycles.

Strict constraints. We may include further domain-specific constraints into the model, to express certain disallowed configurations. For instance, proposition types that appear in CDCP data can be ordered by the level of objectivity (Park et al., 2015a), as shown in Table 1. In a well-formed argument, we would want to see links from more objective to equally or less objective propositions: it's fine to provide FACT as reason for VALUE, but not the other way around. While the training data sometimes violates this constraint, enforcing it might provide a useful inductive bias.

Inference. The $\arg\max$ in Equation 1 is a MAP over a factor graph with cycles and many overlapping factors, including logic factors. While exact inference methods are generally unavailable, our setting is perfectly suited for the Alternating Directions Dual Decomposition (AD^3) algorithm: approximate inference on expressive factor graphs with overlapping factors, logic constraints, and generic factors (e.g., directed tree factors) defined through maximization oracles (Martins et al., 2015). When AD^3 returns an integral solution, it is globally optimal, but when solutions are fractional, several options are available. At test time, for analysis, we retrieve exact solutions using the branch-and-bound method. At training time, however, fractional solutions can be used *as-is*; this makes better use of each iteration and actually increases the ratio of integral solutions in future iterations, as well as at test time, as proven by Meshi et al. (2016). We also find that after around 15 training iterations with fractional solutions, over 99% of inference calls are integral.

Learning. We train the models by minimizing the structured hinge loss (Taskar et al., 2004):

$$\sum_{(x,y)\in\mathcal{D}} \max_{y'\in\mathcal{Y}} (f(x,y';\boldsymbol{w}) + \rho(y,y')) - f(x,y;\boldsymbol{w}) \tag{2}$$

where ρ is a configurable misclassification cost. The $\max$ in Equation 2 is not the same as the one used for prediction, in Equation 1. However, when the cost function ρ decomposes over the variables, cost-augmented inference amounts to regular inference after augmenting the potentials accordingly. We use a weighted Hamming cost:

$$\rho(y,\hat{y}) := \sum_v \rho(y_v)\mathbb{I}[y_v = \hat{y}_v]$$

where v is summed over all variables in a document $\{a\} \cup \{a \to b\}$, and $\rho(y_v)$ is a misclassification cost. We assign uniform costs ρ to 1 for all mistakes except false-negative links, where we use higher cost proportional to the class imbalance in the training split, effectively giving more weight to positive links during training.

4.3 Argument structure SVM

One option for parameterizing the potentials of the unary and higher-order factors is with *linear models*, using proposition, link, and higher-order features. This gives birth to a linear structured SVM (Tsochantaridis et al., 2005), which, when using l_2 regularization, can be trained efficiently in the dual using the online block-coordinate Frank-Wolfe algorithm of Lacoste-Julien et al. (2013), as implemented in the `pystruct` library (Müller and Behnke, 2014). This algorithm is more convenient than subgradient methods, as it does not require tuning a learning rate parameter.

Features. For unary proposition and link features, we faithfully follow Stab and Gurevych (2016, Tables 9 and 10): proposition features are

[8] A tree factor regards each bound variable as an edge in a graph and assigns $-\infty$ scores to configurations that are not valid trees. For inference, we can use maximum spanning arborescence algorithms such as Chu-Liu/Edmonds.

Model part	CDCP dataset	UKP dataset
proposition types	REFERENCE $\succ$ TESTIMONY $\succ$ FACT $\succ$ VALUE $\succ$ POLICY	CLAIM, MAJOR CLAIM, PREMISE
links	all possible	within each paragraph
2nd order factors	siblings, co-parents	grandparents, co-parents
link structure	transitive acyclic: • $a \to b \;\&\; b \to c \implies a \to c$ • ATMOSTONE$(a \to b, b \to a)$	directed forest: • TREEFACTOR over each paragraph • zero-potential "root" links $a \to *$
strict constraints	link source must be as least as objective as the target: $a \to b \implies a \succeq b$	link source must be premise: $a \to b \implies a =$ PREMISE

Table 1: Instantiation of model design choices for each dataset.

lexical (unigrams and dependency tuples), structural (token statistics and proposition location), indicators (from hand-crafted lexicons), contextual, syntactic (subclauses, depth, tense, modal, and POS), probability, discourse (Lin et al., 2014), and average GloVe embeddings (Pennington et al., 2014). Link features are lexical (unigrams), syntactic (POS and productions), structural (token statistics, proposition statistics and location features), hand-crafted indicators, discourse triples, PMI, and shared noun counts.

Our proposed higher-order factors for grandparent, co-parent, and sibling structures require features extracted from a proposition triplet a, b, c. In dependency and semantic parsing, higher-order factors capture relationships between words, so sparse indicator features can be efficiently used. In our case, since propositions consist of many words, BOW features may be too noisy and too dense; so for simplicity we again take a cue from the link-specific features used by Stab and Gurevych (2016). Our higher-order factor features are: same sentence indicators (for all 3 and for each pair), proposition order (one for each of the 6 possible orderings), Jaccard similarity (between all 3 and between each pair), presence of any shared nouns (between all 3 and between each pair), and shared noun ratios: nouns shared by all 3 divided by total nouns in each proposition and each pair, and shared nouns between each pair with respect to each proposition. Up to vocabulary size difference, our total feature dimensionality is approximately 7000 for propositions and 2100 for links. The number of second order features is 35.

Hyperparameters. We pick the SVM regularization parameter $C \in \{0.001, 0.003, 0.01, 0.03, 0.1, 0.3\}$ by k-fold cross validation at document level, optimizing for the average between link and proposition F_1 scores.

4.4 Argument structure RNN

Neural network methods have proven effective for natural language problems even with minimal-to-no feature engineering. Inspired by the use of LSTMs (Hochreiter and Schmidhuber, 1997) for MST dependency parsing by Kiperwasser and Goldberg (2016), we parametrize the potentials in our factor graph with an LSTM-based neural network,[9] replacing MST inference with the more general AD^3 algorithm, and using relaxed solutions for training when inference is inexact.

We extract embeddings of all words with a corpus frequency > 1, initialized with GloVe word vectors. We use a deep bidirectional LSTM to encode contextual information, representing a proposition a as the average of the LSTM outputs of its words, henceforth denoted $\overleftrightarrow{\boldsymbol{a}}$.

Proposition potentials. We apply a multi-layer perceptron (MLP) with rectified linear activations to each proposition, with all layer dimensions equal except the final output layer, which has size $|\mathcal{P}|$ and is not passed through any nonlinearities.

Link potentials. To score a dependency $a \to b$, Kiperwasser and Goldberg (2016) pass the concatenation $[\overleftrightarrow{\boldsymbol{a}}; \overleftrightarrow{\boldsymbol{b}}]$ through an MLP. After trying this, we found slightly better performance by first passing *each* proposition through a slot-specific dense layer $\left(\overline{\boldsymbol{a}} := \sigma_{\text{src}}(\overleftrightarrow{\boldsymbol{a}}), \overline{\boldsymbol{b}} := \sigma_{\text{trg}}(\overleftrightarrow{\boldsymbol{b}})\right)$ followed by a *bilinear* transformation:

$$\phi_{\text{on}}(a \to b) := \overline{\boldsymbol{a}}^{\top} \boldsymbol{W} \overline{\boldsymbol{b}} + \boldsymbol{w}_{\text{src}}^{\top} \overline{\boldsymbol{a}} + \boldsymbol{w}_{\text{trg}}^{\top} \overline{\boldsymbol{b}} + w_0^{(\text{on})}.$$

Since the bilinear expression returns a scalar, but the link potentials must have a value for both the `on` and `off` states, we set the full potential to $\boldsymbol{\phi}(a \to b) := [\phi_{\text{on}}(a \to b), w_0^{(\text{off})}]$ where $w_0^{(\text{off})}$ is a learned scalar bias. We initialize $\boldsymbol{W}$ to the diagonal identity matrix.

[9] We use the `dynet` library (Neubig et al., 2017).

Second order potentials. Grandparent potentials $\phi(a \to b \to c)$ score two adjacent directed edges, in other words three propositions. We again first pass each proposition representation through a slot-specific dense layer. We implement a multilinear scorer analogously to the link potentials:

$$\phi(a \to b \to c) := \sum_{i,j,k} \overline{\boldsymbol{a}}_i \overline{\boldsymbol{b}}_j \overline{\boldsymbol{c}}_k w_{ijk}$$

where $\mathcal{W} = (w)_{ijk}$ is a third-order cube tensor. To reduce the large numbers of parameters, we implicitly represent $\mathcal{W}$ as a rank r tensor: $w_{ijk} = \sum_{s=1}^{r} u_{is}^{(1)} u_{js}^{(2)} u_{ks}^{(3)}$. Notably, this model captures only third-order interactions between the representation of the three propositions. To capture first-order "bias" terms, we could include slot-specific linear terms, e.g., $\boldsymbol{w}_a^\top \overline{\boldsymbol{a}}$; but to further capture quadratic *backoff* effects (for instance, if two propositions carry a strong signal of being siblings regardless of their parent), we would require quadratically many parameters. Instead of explicit lower-order terms, we propose *augmenting* $\overline{\boldsymbol{a}}, \overline{\boldsymbol{b}}$, and $\overline{\boldsymbol{c}}$ with a constant feature of 1, which has approximately the same effect, while benefiting from the parameter sharing in the low-rank factorization; an effect described by Blondel et al. (2016). Siblings and co-parents factors are similarly parametrized with their own tensors.

Hyperparameters. We perform grid search using k-fold document-level cross-validation, tuning the dropout probability in the dense MLP layers over $\{0.05, 0.1, 0.15, 0.2, 0.25\}$ and the optimal number of passes over the training data over $\{10, 25, 50, 75, 100\}$. We use 2 layers for the LSTM and the proposition classifier, 128 hidden units in all layers, and a multilinear decomposition with rank $r = 16$, after preliminary CV runs.

4.5 Baseline models

We compare our proposed models to equivalent independent unary classifiers. The unary-only version of a structured SVM is an l_2-regularized linear SVM.[10] For the RNN, we compute unary potentials in the same way as in the structured model, but apply independent hinge losses at each variable, instead of the global structured hinge loss. Since the RNN weights are shared, this is a form of multi-task learning. The baseline predictions can be interpreted as unary potentials, therefore we can simply round their output to the highest scoring labels, or we can, alternatively, perform test-time inference, imposing the desired structure.

5 Results

We evaluate our proposed models on both datasets. For model selection and development we used k-fold cross-validation at document level: on CDCP we set $k = 3$ to avoid small validation folds, while on UKP we follow Stab and Gurevych (2016) setting $k = 5$. We compare our proposed structured learning systems (the linear structured SVM and the structured RNN) to the corresponding baseline versions. We organize our experiments in three incremental variants of our factor graph: *basic*, *full*, and *strict*, each with the following components:[11]

component	basic	full	strict	(baseline)
unaries	✓	✓	✓	✓
compat. factors	✓	✓	✓	
compat. features		✓	✓	
higher-order		✓	✓	
link structure		✓	✓	✓
strict constraints			✓	✓

Following Stab and Gurevych (2016), we compute F_1 scores at proposition and link level, and also report their average as a summary of overall performance.[12] The results of a single prediction run on the test set are displayed in Table 2. The overall trend is that training using a structured objective is better than the baseline models, even when structured inference is applied on the baseline predictions. On UKP, for link prediction, the linear baseline can reach good performance when using inference, similar to the approach of Stab and Gurevych (2016), but the improvement in proposition prediction leads to higher overall F_1 for the structured models. Meanwhile, on the more difficult CDCP setting, performing inference on the baseline output is not competitive. While feature engineering still outperforms our RNN model, we find that RNNs shine on proposition classification, especially on UKP, and that structured training can make them competitive, reducing their observed lag on link prediction (Katiyar and Cardie, 2016), possibly through mitigating class imbalance.

[10] We train our SVM using SAGA (Defazio et al., 2014) in `lightning` (Blondel and Pedregosa, 2016).

[11] Components are described in Section 4. The baselines *with inference* support only unaries and factors with no parameters, as indicated in the last column.

[12] For link F_1 scores, however, we find it more intuitive to only consider retrieval of positive links rather than macro-averaged two-class scores.

	Baseline						Structured					
	SVM			RNN			SVM			RNN		
Metric	basic	full	strict	basic	full	strict	basic	full	strict	basic	full	strict
CDCP dataset												
Average	47.4	47.3	47.9	40.8	38.0	38.0	48.1	49.3	**50.0**	43.5	33.5	38.2
Link (272)	22.0	21.9	23.8	9.9	12.8	12.8	24.7	25.1	**26.7**	14.4	14.6	10.5
Proposition	72.7	72.7	72.0	71.8	63.2	63.2	71.6	**73.5**	73.2	72.7	52.4	65.9
VALUE (491)	75.3	75.3	74.4	74.1	74.8	74.8	73.4	75.7	76.4	73.7	73.1	69.7
POLICY (153)	78.7	78.7	78.5	74.3	72.2	72.2	72.3	77.3	76.8	73.9	74.4	76.8
TESTIMONY (204)	70.3	70.3	68.6	74.6	71.8	71.8	69.8	71.7	71.5	74.2	72.3	75.8
FACT (124)	39.2	39.2	38.3	35.8	30.5	30.5	42.4	42.5	41.3	41.5	42.2	40.5
REFERENCE (1)	100.0	100.0	100.0	100.0	66.7	66.7	100.0	100.0	100.0	100.0	0.0	66.7
UKP dataset												
Average	64.7	66.6	66.5	58.7	57.4	58.7	67.1	**68.9**	67.1	59.0	63.6	64.7
Link (809)	55.8	59.7	**60.3**	44.8	43.8	44.0	56.9	60.1	56.9	44.1	50.4	50.1
Proposition	73.5	73.5	72.6	72.6	70.9	73.3	77.2	77.6	77.3	74.0	76.9	**79.3**
MAJOR CLAIM (153)	76.7	76.7	77.6	81.4	75.1	81.3	77.0	78.2	80.0	83.6	84.6	88.3
CLAIM (304)	55.4	55.4	52.0	51.7	52.7	53.5	64.3	64.5	62.8	53.2	60.2	62.0
PREMISE (809)	88.4	88.4	88.3	84.8	84.8	85.2	90.3	90.2	89.2	85.0	85.9	87.6

Table 2: Test set F_1 scores for link and proposition classification, as well as their average, on the two datasets. The number of test instances is shown in parentheses; best scores on overall tasks are in bold.

5.1 Discussion and analysis

Contribution of compatibility features. The compatibility factor in our model can be visualized as conditional odds ratios given the source and target proposition types. Since there are only four possible configurations of the compatibility features, we can plot all cases in Figure 3, alongside the basic model. Not using compatibility features, the basic model can only learn whether certain configurations are more likely than others (e.g. a REFERENCE supporting another REFERENCE is unlikely, while a REFERENCE supporting a FACT is more likely; essentially a soft version of our domain-specific strict constraints. The full model with compatibility features is finer grained, capturing, for example, that links from REFERENCE to FACT are more likely when the reference comes *after*, or that links from VALUE to POLICY are extremely likely only when the two are adjacent.

Proposition errors. The confusion matrices in Figure 4 reveal that the most common confusion is misclassifying FACT as VALUE. The strongest difference between the various models tested is that the RNN-based models make this error less often. For instance, in the proposition:

> And the single most frequently used excuse of any debtor is "I didn't receive the letter/invoice/statement"

the pronouns in the nested quote may be mistaken for subjectivity, leading to the structured SVMs predictions of VALUE or TESTIMONY, while the basic structured RNN correctly classifies it as FACT.

Link errors. While structured inference certainly helps baselines by preventing invalid structures such as cycles, it still depends on local decisions, losing to fully structured training in cases where joint proposition and link decisions are needed. For instance, in the following conclusion of an UKP essay, the annotators found no links:

> In short, [the individual should finance his or her education]$_a$ because [it is a personal choice.]$_b$ Otherwise, [it would cause too much cost from taxpayers and the government.]$_c$

Indeed, no reasons are provided, but baseline are misled by the connectives: the SVM baseline outputs that b and c are PREMISES supporting the CLAIM a. The full structured SVM combines the two tasks and correctly recognizes the link structure.

Linear SVMs are still a very good baseline, but they tend to overgenerate links due to class imbalance, even if we use class weights during training. Surprisingly, RNNs are at the opposite end, being extremely conservative, and getting the highest precision among the models. On CDCP, where the number of true links is 272, the linear baseline with strict inference predicts 796 links with a precision of only 16%, while the strict structured RNN only predicts 52 links, with 33% precision; the example in Figure 5 illustrates this. In terms of higher-order structures, we find that using higher-order factors increases precision, at a cost in recall.

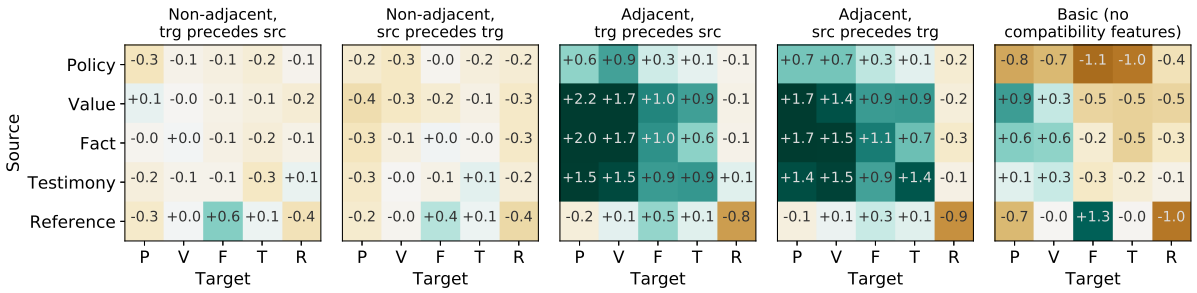


Figure 3: Learned conditional log-odds $\log \frac{p(\texttt{on}|\cdot)}{p(\texttt{off}|\cdot)}$, given the source and target proposition types and compatibility feature settings. First four figures correspond to the four possible settings of the compatibility features in the full structured SVM model. For comparison, the rightmost figure shows the same parameters in the basic structured SVM model, which does not use compatibility features.

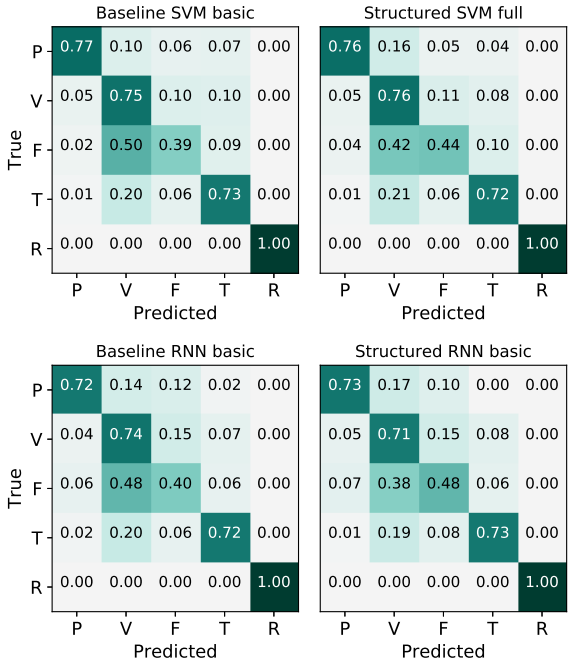


Figure 4: Normalized confusion matrices for proposition type classification.

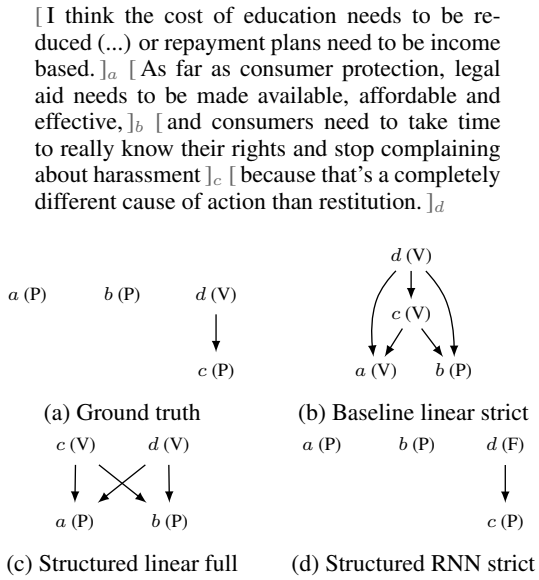


Figure 5: Predictions on a CDCP comment where the structured RNN outperforms the other models.

This is most beneficial for the 856 co-parent structures in the UKP test set: the full structured SVM has 53% F_1, while the basic structured SVM and the basic baseline get 47% and 45% respectively. On CDCP, while higher-order factors help, performance on siblings and co-parents is below 10% F_1 score. This is likely due to link sparsity and suggests plenty of room for further development.

6 Conclusions and future work

We introduce an argumentation parsing model based on AD^3 relaxed inference in expressive factor graphs, experimenting with both linear structured SVMs and structured RNNs, parametrized with higher-order factors and link structure constraints. We demonstrate our model on a new argumentation mining dataset with more permissive argument structure annotation. Our model also achieves state-of-the-art link prediction performance on the UKP essays dataset.

Future work. Stab and Gurevych (2016) found polynomial kernels useful for modeling feature interactions, but kernel structured SVMs scale poorly, we intend to investigate alternate ways to capture feature interactions. While we focus on monological argumentation, our model could be extended to dialogs, for which argumentation theory thoroughly motivates non-tree structures (Afantenos and Asher, 2014).

Acknowledgements

We are grateful to André Martins, Andreas Müller, Arzoo Katyiar, Chenhao Tan, Felix Wu, Jack Hessel, Justine Zhang, Mathieu Blondel, Tianze Shi, Tobias Schnabel, and the rest of the Cornell NLP seminar for extremely helpful discussions. We thank the anonymous reviewers for their thorough and well-argued feedback.

References

Stergos Afantenos and Nicholas Asher. 2014. Counter-argumentation and discourse: A case study. In *Proceedings of ArgNLP*.

Mathieu Blondel, Masakazu Ishihata, Akinori Fujino, and Naonori Ueda. 2016. Polynomial networks and factorization machines: New insights and efficient training algorithms. In *Proceedings of ICML*.

Mathieu Blondel and Fabian Pedregosa. 2016. Lightning: large-scale linear classification, regression and ranking in Python. https://doi.org/10.5281/zenodo.200504.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Valentin Tablan, Niraj Aswani, Ian Roberts, Genevieve Gorrell, Adam Funk, Angus Roberts, Danica Damljanovic, Thomas Heitz, Mark A. Greenwood, Horacio Saggion, Johann Petrak, Yaoyong Li, and Wim Peters. 2011. *Text Processing with GATE (Version 6)*.

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. 2014. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Proceedings of NIPS*.

Ivan Habernal and Iryna Gurevych. 2016. Argumentation mining in user-generated web discourse. *Computational Linguistics* .

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9(8):1735–1780.

Arzoo Katiyar and Claire Cardie. 2016. Investigating LSTMs for joint extraction of opinion entities and relations. In *Proceedings of ACL*.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *arXiv:1603.04351* preprint.

Klaus Krippendorff. 1980. *Content Analysis: An Introduction to Its Methodology*. Commtext. Sage.

Frank R Kschischang, Brendan J Frey, and H-A Loeliger. 2001. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory* 47(2):498–519.

Simon Lacoste-Julien, Martin Jaggi, Mark Schmidt, and Patrick Pletscher. 2013. Block-coordinate Frank-Wolfe optimization for structural SVMs. In *Proceedings of ICML*.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2014. A PDTB-styled end-to-end discourse parser. *Natural Language Engineering* 20(02):151–184.

André FT Martins and Mariana SC Almeida. 2014. Priberam: A Turbo Semantic Parser with second order features. In *Proceedings of SemEval*.

André FT Martins, Miguel B Almeida, and Noah A Smith. 2013. Turning on the Turbo: Fast third-order non-projective Turbo Parsers. In *Proceedings of ACL*.

André FT Martins, Mário AT Figueiredo, Pedro MQ Aguiar, Noah A Smith, and Eric P Xing. 2015. AD3: Alternating directions dual decomposition for MAP inference in graphical models. *Journal of Machine Learning Research* 16:495–545.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of EMNLP*.

Ofer Meshi, Mehrdad Mahdavi, Adrian Weller, and David Sontag. 2016. Train and test tightness of LP relaxations in structured prediction. In *Proceedings of ICML*.

Andreas C Müller and Sven Behnke. 2014. PyStruct: learning structured prediction in Python. *Journal of Machine Learning Research* 15(1):2055–2060.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, Kevin Duh, Manaal Faruqui, Cynthia Gan, Dan Garrette, Yangfeng Ji, Lingpeng Kong, Adhiguna Kuncoro, Gaurav Kumar, Chaitanya Malaviya, Paul Michel, Yusuke Oda, Matthew Richardson, Naomi Saphra, Swabha Swayamdipta, and Pengcheng Yin. 2017. DyNet: The dynamic neural network toolkit. *arXiv:1701.03980* preprint.

Joonsuk Park, Cheryl Blake, and Claire Cardie. 2015a. Toward machine-assisted participation in eRulemaking: An argumentation model of evaluability. In *Proceedings of ICAIL*.

Joonsuk Park, Arzoo Katiyar, and Bishan Yang. 2015b. Conditional random fields for identifying appropriate types of support for propositions in online user comments. In *Proceedings of the 2nd Workshop on Argumentation Mining*. Association for Computational Linguistics, Denver, CO, pages 39–44.

Andreas Peldszus and Manfred Stede. 2015. Joint prediction in MST-style discourse parsing for argumentation mining. In *Proceedings of EMNLP*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of EMNLP*.

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2016. Here's my point: Argumentation mining with pointer networks. *arXiv:1612.08994* preprint.

Christian Stab and Iryna Gurevych. 2016. Parsing argumentation structures in persuasive essays. *arXiv:1604.07370* preprint.

Ben Taskar, Carlos Guestrin, and Daphne Koller. 2004. Max-margin Markov networks. In *Proceedings of NIPS*.

Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research* 6(Sep):1453–1484.