Reducing Distant Supervision Noise with Maxpooled Attention and Sentence-Level Supervision

Anonymous EMNLP submission

Abstract

We propose an effective multitask learning setup for reducing distant supervision noise by leveraging sentence-level supervision. We show how sentence-level supervision can be used to improve the encoding of individual sentences, and to learn which input sentences are more likely to express the relationship between a pair of entities. We also introduce a novel neural architecture for collecting signals from multiple input sentences, which combines the benefits of attention and maxpooling. The proposed method increases AUC by 10% (from 0.261 to 0.284), and outperforms recently published results on the FB-NYT dataset.

1 Introduction

Early work in relation extraction from text used fully supervised methods, e.g., Bunescu and Mooney (2005), which motivated the development of relatively small datasets with sentence-level annotations such as ACE 2004/2005, BioInfer and SemEval 2010 Task 8. Recognizing the difficulty of annotating text with relations, especially when the number of relation types of interest is large, Mintz et al. (2009) pioneered the distant supervision approach to relation extraction, where a knowledge base (KB) and a text corpus are used to automatically generate a large dataset of labeled sentences which is then used to train a relation classifier. Distant supervision provides a practical alternative to manual annotations, but introduces many noisy examples. Although many methods have been proposed to reduce the noise in distantly supervised models for relation extraction (e.g., Hoffmann et al., 2011; Surdeanu et al., 2012; Roth et al., 2013; Fan et al., 2014; Zeng et al., 2015; Jiang et al., 2016; Liu et al., 2017), a rather obvious approach has been understudied: using sentence-level supervision to augment distant supervision. Intuitively, supervision at the sentence-level can help reduce the noise in distantly supervised models by identifying which of the input sentences for a given pair of entities are likely to express a relation.

We experiment with a variety of model architectures to combine sentence- and bag-level supervision and find it most effective to use the sentence-level annotations to directly supervise the sentence encoder component of the model in a multi-task learning framework. We also introduce a novel maxpooling attention architecture for combining the evidence provided by different sentences where the entity pair is mentioned, and use the sentence-level annotations to supervise attention weights.

The contributions of this paper are as follows:

- We propose an effective multitask learning setup for reducing distant supervision noise by leveraging existing datasets of relations annotated at the sentence level.
- We propose *maxpooled attention*, a neural architecture which combines the benefits of maxpooling and soft attention, and show that it helps the model combine information about a pair of entities from multiple sentences.
- We release our library for relation extraction as open source.[1]

The following section defines the notation we use, describes the problem and provides an overview of our approach.

2 Overview

Our goal is to predict which relation types are expressed between a pair of entities (e_1, e_2), given

[1] We attach the anonymized code as supplemental material in this submission instead of providing a github link in order to maintain author anonymity.

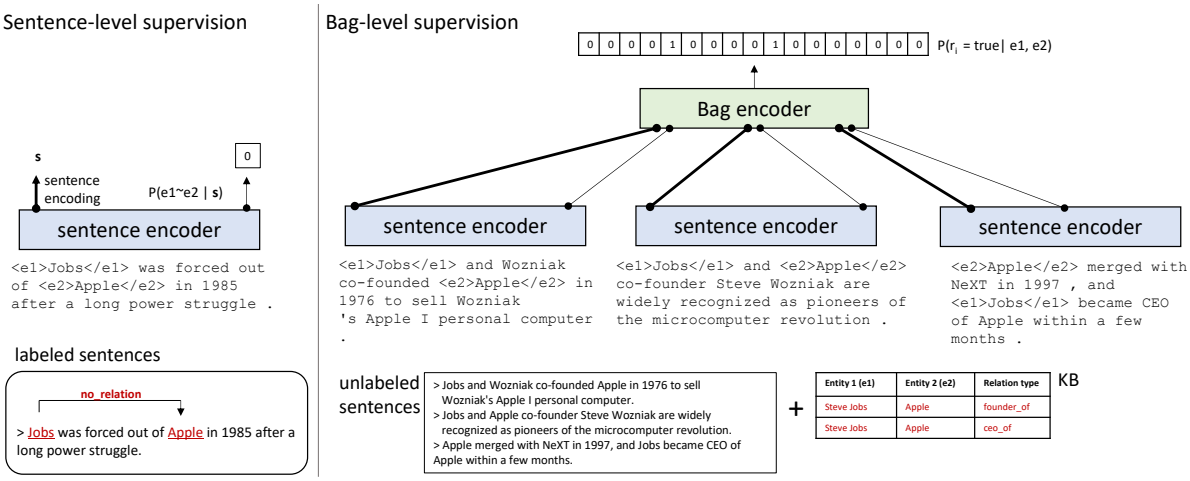

Figure 1: An overview of our approach for augmenting distant supervision with sentence-level annotations. The left side shows one sentence in the labeled data and how it is used to provide direct supervision for the sentence encoder. The right side shows snippets of the text corpus and the knowledge base, which are then combined to construct one training instance for the model, with a bag of three input sentences and two active relations: 'founder_of' and 'ceo_of'.

all sentences in which both entities are mentioned in a large collection of unlabeled documents.

Following previous work on distant supervision, we use known tuples (e_1, r, e_2) in a knowledge base $\mathcal{K}$ to automatically annotate sentences where both entities are mentioned. In particular, we group all sentences s with one or more mentions of an entity pair e_1 and e_2 into a bag of sentences B_{e_1,e_2}, then automatically annotate this bag with the set of relation types $L^{\text{distant}} = \{r \in \mathcal{R} : (e_1, r, e_2) \in \mathcal{K}\}$, where $\mathcal{R}$ is the set of relations we are interested in. We use 'positive instances' to refer to cases where $|L| > 0$, and 'negative instances' when $|L| = 0$.

In this paper, we leverage existing datasets with sentence-level relation annotations in a similar domain, where each example consists of a token sequence s, token indexes for e_1 and e_2 in the sequence, and one relation type (or 'no relation'). Since the relation types annotated at the sentence level may not correspond one-to-one to those in the KB, we replace the relation label associated with each sentence with a binary indicator. (1 indicates that the sentence s expresses one of the relationships of interest.) We do not require the entities to match those in the KB either.

Fig. 1 illustrates how we modify neural architectures commonly used in distant supervision, e.g., Lin et al. (2016); Liu et al. (2017) to effectively incorporate sentence-level supervision. The model consists of two components: 1) A **sentence encoder** (displayed in blue) reads a sequence of tokens and their relative distances from e_1 and e_2, and outputs a vector $\mathbf{s}$ representing the sentence encoding, as well as $P(e_1 \sim e_2 \mid \mathbf{s})$ representing the probability that the two entities are related given this sentence. 2) The **bag encoder** (displayed in green) reads the encoding of each sentence in the bag for the pair (e_1, e_2) and predicts $P(r = 1 \mid e_1, e_2), \forall r \in \mathcal{R}$.

We combine both bag-level (i.e., distant) and sentence-level (i.e., direct) supervision in a multi-task learning framework by minimizing the weighted sum of the cross entropy losses for $P(e_1 \sim e_2 \mid \mathbf{s})$ and $P(r = 1 \mid e_1, e_2)$. By sharing the parameters of sentence encoders used to compute either loss, the sentence encoders become less susceptible to the noisy bag labels. The bag encoder also benefits from sentence-level supervision by using the supervised distribution $P(e_1 \sim e_2 \mid \mathbf{s})$ to decide the weight of each sentence in the bag, using a novel architecture which we call *maxpooled attention*.

3 Model

The model predicts a set of relation types $L^{\text{pred}} \subset \mathcal{R}$ given a pair of entities e_1, e_2 and a bag of sentences B_{e_1,e_2}. In this section, we first describe the sentence encoder part of the model (Figure 2, bottom), then describe the bag encoder (Figure 2, top), then we explain how the two types of supervision are jointly used for training the model end-to-end.

3.1 Sentence Encoder Architecture

Given a sequence of words $w_1, \ldots, w_{|s|}$ in a sentence s, a sentence encoder translates this sequence into a fixed length vector $\mathbf{s}$.

Input Representation. The input representation is illustrated graphically with a table at the bottom of Figure 2. We map word token i in the sentence w_i to a pretrained word embedding vector $\mathbf{w}_i$.[2] Another crucial input signal is the position of entity mentions in each sentence $s \in B_{e_1,e_2}$. Following Zeng et al. (2014), we map the distance between each word in the sentence and the entity mentions[3] to a small vector of learned parameters, namely $\mathbf{d}_i^{e_1}$ and $\mathbf{d}_i^{e_2}$.

Instead of randomly initializing position embeddings with mean = 0, we obtain notable performance improvements by randomly initializing all dimensions of the position embedding for distance d around the mean value d. Intuitively, this makes it easier to learn useful parameters since the embedding of similar distances (e.g., $d = 10$ and $d = 11$) should be similar, without adding hard constraints on how they should be related.

We find that adding a dropout layer with a small probability ($p = 0.1$) before the sentence encoder reduces overfitting and improves the results. To summarize, the input layer for a sentence s is a sequence of vectors:

$$\mathbf{v}_i = [\mathbf{w}_i; \mathbf{d}_i^{e_1}; \mathbf{d}_i^{e_2}], \text{ for } i \in 1, \ldots, |s|$$

Word Composition. Word composition is illustrated with the block `CNN` in the bottom part of Figure 2, which represents a convolutional neural network (CNN) with multiple filter sizes. The outputs of the maxpool operations for different filter sizes are concatenated then projected into a smaller vector using one feed forward linear layer.

This is in contrast to previous work (Pennington et al., 2014) which used Piecewise CNN (PCNN). In PCNN, we convolve three segments of the sentence separately: windows before the left entity, windows inbetween the two entities and windows after the right entity. Every split is maxpooled independently, then the three vectors are concatenated. The intuition is that this helps the model put more emphasis on the middle segment which connects the two entities. As discussed later in Section 4.2, we compare CNN and PCNN and find the simpler CNN architecture works better.

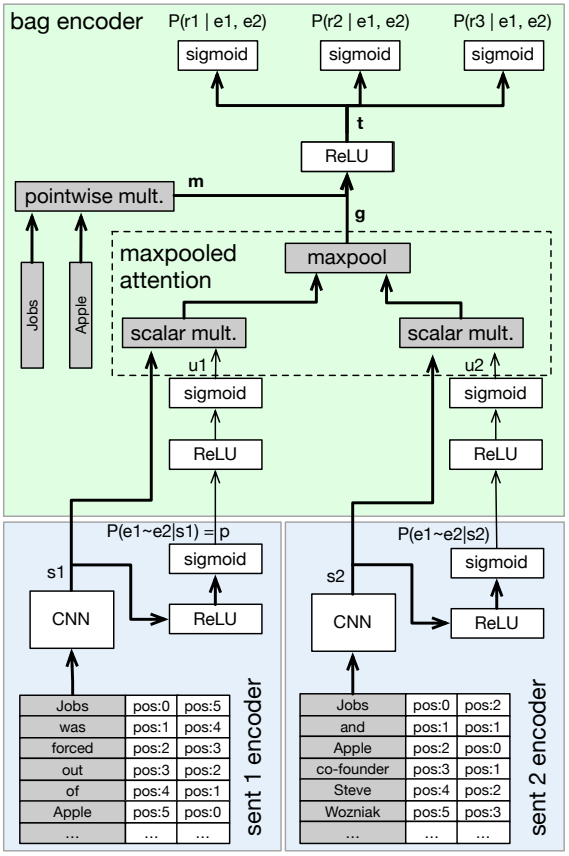

Figure 2: Blue box is the sentence encoder, it maps a sentence to a fixed length vector (`CNN` output) and the probability it expresses a relation between e_1 and e_2 (`sigmoid` output). Green box is the bag encoder, it takes encoded sentences and their weights and produces a fixed length vector (`maxpool` output), concatenates it with entity embeddings (`pointwise mult.` output) then outputs a probability for each relation type r. White boxes contain parameters that the model learns while gray boxes do not have learnable parameters. Sentence-level annotations supervise $P(e_1 \sim e_2 \mid \mathbf{s})$. Bag-level annotations supervise $P(r = 1 \mid e_1, e_2)$.

Sentence encoding $\mathbf{s}$ is computed as follows:

$$\mathbf{c}_x = \text{CNN}_x(\mathbf{v}_1, \ldots, \mathbf{v}_{|s|}), \text{for } x \in \{2, 3, 4, 5\}$$
$$\mathbf{s} = \mathbf{W}_1 [\mathbf{c}_2; \mathbf{c}_3; \mathbf{c}_4; \mathbf{c}_5] + \mathbf{b}_1,$$

where CNN_x is a standard convolutional neural network with filter size x, $\mathbf{W}_1$ and $\mathbf{b}_1$ are model parameters and $\mathbf{s}$ is the sentence encoding.

We feed the sentence encoding s into a ReLU layer followed by a sigmoid layer with output size 1, representing $P(e_1 \sim e_2 \mid \mathbf{s})$, as illustrated in

[2] Following Lin et al. (2016), we do not update the word embeddings while training the model.

[3] If an entity is mentioned more than once in the sentence, we use the distance from the word to the closest entity mention. Distances greater than 30 are mapped to the embedding for distance = 30.

Figure 2 (bottom):

$$P(e_1 \sim e_2 \mid \mathbf{s}) = \quad (1)$$
$$p = \sigma(\mathbf{W}_3 \text{ReLU}(\mathbf{W}_2 \mathbf{s} + \mathbf{b}_2) + \mathbf{b}_3),$$

where σ is the sigmoid function and $\mathbf{W}_2, \mathbf{b}_2, \mathbf{W}_3, \mathbf{b}_3$ are model parameters.

3.2 Bag Encoder Architecture

Given a bag B_{e_1,e_2} of $n \geq 1$ sentences, we compute their encodings $\mathbf{s}_1, \ldots, \mathbf{s}_n$ as described earlier and feed them into the bag encoder, which is responsible for combining the information in all sentence encodings and predict the probability $P(r = 1 \mid e_1, e_2), \forall r \in \mathcal{R}$. The bag encoder also makes use of $p = P(e_1 \sim e_2 \mid \mathbf{s})$ from Eq. 1 as an estimate of the degree to which sentence s expresses the relation between e_1 and e_2.

Maxpooled Attention. To aggregate the sentence encodings $\mathbf{s}_1, \ldots, \mathbf{s}_n$ into a fixed length vector that captures the important features in the bag, Jiang et al. (2016) used maxpooling, while Lin et al. (2016) used soft attention.

In this work, we propose *maxpooled attention*, a new form of attention which combines some of the characteristics of maxpooling and soft attention. Given the encoding $\mathbf{s}_j$ and an unnormalized weight u_j for each sentence $s_j \in B_{e_1,e_2}$, the bag encoding $\mathbf{g}$ is a vector with the same dimensionality as $\mathbf{s}_j$ with the k-th element computed as:

$$\mathbf{g}_j[k] = \max_{j \in 1,\ldots,n} \{\mathbf{s}_j[k] \times \sigma(u_j)\}.$$

Maxpooled attention has the same intuition of soft attention; learning weights for sentences that enable the model to focus on the important sentences. However, maxpooled attention differs from soft attention in two aspects.

The first is that every sentence s_j is given a probability that indicates how useful the sentence is, independently of the other sentences. Notice how this is different from soft attention where sentences compete for probability mass, i.e., probabilities must sum to 1. This is implemented in maxpooled attention by normalizing the weight of each sentence with a sigmoid function rather than a softmax. This is a better fit for the task at hand because the sentences are not competing. It also makes the weights useful even when $|B_{e_1,e_2}| = 1$, while soft attention will always normalize such weights to 1.

The second difference between maxpooled attention and soft attention is the use of weighted maxpooling instead of weighted average. Maxpooling is more effective for this task because it can pick the useful features from different sentences.

As shown in Figure 2, we do not directly use the p from Eq. 1 as weight in maxpooled attention. Instead, we found it useful to feed it into more non-linearities. The unnormalized maxpooled attention weight for s_j is computed as:

$$u_j = \mathbf{W}_7 \text{ ReLU}(\mathbf{W}_6 \, p + \mathbf{b}_6) + \mathbf{b}_7.$$

Entity Embeddings. Following Ji et al. (2017), we use entity embeddings to improve our model of relations in the distant supervision setting, although our formulation is closer to that of Yang et al. (2015) who used point-wise multiplication of entity embeddings: $\mathbf{m} = \mathbf{e}_1 \odot \mathbf{e}_2$, where $\odot$ is point-wise multiplication, and $\mathbf{e}_1$ and $\mathbf{e}_2$ are the embeddings of e_1 and e_2, respectively. In order to improve the coverage of entity embeddings, we use pretrained GloVe vectors (Pennington et al., 2014) (same embeddings used in the input layer). For entities with multiple words, like "Steve Jobs", the vector for the entity is the average of the GloVe vectors of its individual words. If the entity is expressed differently across sentences, we average the vectors of the different mentions. As discussed in Section 4.2, this leads to big improvement in the results, and we believe there's still big room for improvement from having better representation for entities. We feed the output $\mathbf{m}$ as additional input to the last block of our model.

Output Layer. The final step is to use the bag encoding $\mathbf{g}$ and the entity pair encoding $\mathbf{m}$ to predict a set of relations L^{pred} which is a standard multilabel classification problem. We concatenate $\mathbf{g}$ and $\mathbf{m}$ and feed them into a feedforward layer with ReLU non-linearity, followed by a sigmoid layer with an output size of $|\mathcal{R}|$:

$$\mathbf{t} = \text{ReLU}(\mathbf{W}_4[\mathbf{g}; \mathbf{m}] + \mathbf{b}_4)$$
$$P(\mathbf{r} = \mathbf{1} \mid e_1, e_2) = \sigma(\mathbf{W}_5 \mathbf{t} + \mathbf{b}_5),$$

where $\mathbf{r}$ is a vector of Bernoulli variables each of which corresponds to one of the relations in $\mathcal{R}$. This is the final output of the model.

3.3 Model Training

To train the model on the bag-level labels obtained with distant supervision, we use binary cross-entropy loss between the model predictions and

the labels obtained with distant supervision, i.e.,

$$\text{bag\_loss} = \sum_{B_{e_1,e_2}} -\log P(\mathbf{r} = \mathbf{r}^{\text{distant}} \mid e_1, e_2)$$

where $\mathbf{r}^{\text{distant}}[k] = 1$ indicates that the tuple (e_1, r_k, e_2) is in the knowledge base.

In addition to the bag-level supervision commonly used in distant supervision, we also use sentence-level annotations. One approach is to create a bag of size 1 for each sentence-level annotation, and add the bags to those obtained with distant supervision. However, this approach requires mapping relations in the sentence-level annotations map to those in the KB.

Instead, we found that the best use of the supervised data is to improve the model's ability to predict the the potential usefulness of a sentence by using sentence-level annotations to help supervise the sentence encoder module. According to our analysis of baseline models, distinguishing between positive and negative examples is the real bottleneck. This supervision serves two purposes: it improves our encoding of each sentence, and improves the weights used by the maxpooled attention to decide which sentences should contribute more to the bag encoding.

We maximize log loss of gold labels in the sentence-level data $\mathcal{D}$ according to the model described in Eq. 1:

$$\text{sent\_loss} = \sum_{s, l^{\text{gold}} \in \mathcal{D}} -\log P(l = l^{\text{gold}} \mid \mathbf{s}) \quad (2)$$

where $\mathcal{D}$ consists of all the sentence-level annotations in addition to all distantly-supervised *negative* examples.[4]

We jointly train the model on both types of supervision. The model loss is a weighted sum of the sentence-level and the bag-level losses:

$$\text{loss} = \frac{1}{\lambda + 1} \times \text{bag\_loss} + \frac{\lambda}{\lambda + 1} \times \text{sent\_loss}$$

where λ is a parameter that controls the contribution of each loss, tuned on a validation set.

4 Experiments

4.1 Data and Setup

This section discusses datasets, metrics, experiment configurations and the models we are comparing with.

[4]We note that the distantly supervised negative examples may still be noisy. However, given that relations tend to be sparse, the noise to signal ratio is high.

Distantly Supervised Dataset. The FB-NYT dataset[5] introduced in Riedel et al. (2010) was generated by aligning Freebase facts with The New York Times articles. They used the articles of *2005* and *2006* for training, and *2007* for testing. Recent prior work (Lin et al., 2016; Liu et al., 2017; Huang and Wang, 2017) changed the original dataset and trained on articles except *2007* which were left for testing as in Riedel et al. (2010). We use the modified dataset which was made available by Lin et al. (2016).[6]

	Train	Test
Positive bags	16,625	1,950
Negative bags	236,811	94,917
Sentences	472,963	172,448

Fully Supervised Dataset. We get the sentence-level supervision from the dataset by Angeli et al. (2014) which was collected within their active learning framework. We use sentences with the relevant relations, which results in a dataset consisting of 17,291 positive examples and 11,049 negative examples. It is important to mention that there's no overlap between the test set and the labeled examples in this dataset.[7]

Metrics. Prior works used precision-recall (PR) curves to show their results on the FB-NYT dataset. In this multilabel classification setting, all model predictions for all relation types are sorted by confidence from highest to lowest. Then applying different thresholds gives the points on the PR curve. We use the area under the PR curve (AUC) for early stopping and hyperparameter tuning.

Because we are interested in the high-precision extractions, we focus on the high-precision low-recall part of the PR curve. That is, in our experiments, we only keep points on the PR curve with recall below 0.4 which means that the largest possible value for AUC is 0.4.

Configurations. The FB-NYT dataset does not have a validation set for hyper-parameter tuning and early stopping. For these, Liu et al. (2017) use the test set, and Lin et al. (2016) use 3-fold cross validation. We use 90% of the training set for training and keep the other 10% for validation.

The pretrained word embeddings we use are 300-dimensional GloVe vectors, trained on 42B tokens. Since we do not update word embeddings

[5]http://iesl.cs.umass.edu/riedel/ecml/

[6]https://github.com/thunlp/NRE

[7]We will make this dataset available with the final draft.

while training the model, our vocabulary may include any word which appears in the training, validation or test sets with frequency greater than two. When a word with a hyphen (e.g., 'five-star') is not in the GloVe vocabulary, we average the embeddings of its subcomponents. Otherwise, all OOV words are assigned the same random vector (normal with mean 0 and standard deviation 0.05).

Our model is implemented using PyTorch and AllenNLP (Gardner et al., 2017) and trained on machines with P100 GPUs. Each run takes five hours on average. We train for a large number of epochs and use early stopping with patience $= 3$. The batches of the two datasets are randomly shuffled before every epoch. The optimizer we use is Adam with its default PyTorch parameters. We run every configuration three times with three different random seeds then report the PR curve for the run with the best validation (thresholded) AUC. In the controlled experiments, we report the mean and standard deviation of the AUC metric.

Compared Models. Our baseline for comparison is a model that is similar to what is described in Section 3 with the following configurations. It uses our approach for position embedding initialization, encodes sentences using CNN, uses entity embeddings, aggregate sentences using maxpooling and does not use the sentence-level annotation. Our best configuration adds the maxpooled attention and the sentence-level annotations. We also compare with existing models in the literature. The model by Lin et al. (2016) uses an attention mechanism that assigns weights to each sentence followed by a weighted average of sentence encodings. The model by Liu et al. (2017) extends the model by Lin et al. (2016) by using soft labels during training.

4.2 Results

Main Result. Figure 3 summarizes the main results of our experiments.[8] The AUC of our baseline (green) is comparable to that of Lin et al. (2016) (blue), which verifies that we are building on a strong baseline. Adding maxpooled attention and sentence-level supervision (i.e., the full model, in red) substantially improves over the baseline (green). The figure also illustrates that our full model outperforms the strong baseline of (Liu et al., 2017) in orange.[9]

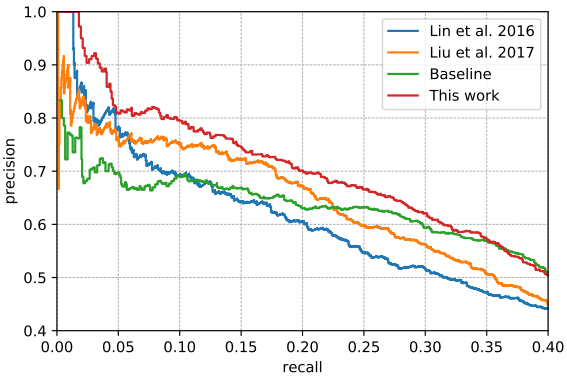

Figure 3: Precision-recall curve comparing our baseline and best configuration with the performance of existing models.

Configurations	AUC
baseline	0.261 ± 0.004
$-$ position embedding init.	0.223 ± 0.031
$+$ PCNN	0.229 ± 0.011
$-$ entity embeddings	0.247 ± 0.002
$+$ attention	0.258 ± 0.014
$+$ maxpooled attention	0.271 ± 0.007
baseline $+$ maxpooled att.	0.271 ± 0.007
$+$ additional bags	0.269 ± 0.001
$+$ sentence loss	$\mathbf{0.284 \pm 0.007}$

Table 1: The $+$ and $-$ signs indicate independent changes to the baseline configuration.

We emphasize that the improved results reported here conflate both additional supervision and model improvements. Next, we report the results of controlled experiments to quantify the contribution of each modification in Table 1. The first line in the table is the baseline model and configuration described in the previous section and in Figure 3, and the $+$ and $-$ signs indicate (independent) additions to and removals from that configuration, respectively.

Position Embedding Initialization. The second line in Table 1 shows that removing the distance-based initialization of position embeddings results in a large drop in AUC. We hypothesize that the position-based initialization reduces the burden of finding optimal values for position embeddings, without explicit constraints that guarantee similar distances to have similar embeddings.

[8] Results of Lin et al. (2016) and Liu et al. (2017) are copied from their papers.

[9] Our results are also competitive with state of the art results in Ye et al. (2017), but we were not able to regenerate the PR curves in their paper.

Sentence Encoder. In the next line of Table 1, we replace the simpler CNN in our baseline with the more complex PCNN (Zeng et al., 2015). Both encoders use filters of sizes 2, 3, 4 and 5. Table 1 shows that using CNN works markedly better than PCNN which is in contrast to the findings of Zeng et al. (2015). This could be due to the use of multiple filter sizes and to the improved representation of entity positions in our model, which may obviate the need to have a separate encoding of each segment in the sentence.

Entity Embeddings. The next line in Table 1 shows that entity embeddings (which are included in the baseline model) provide valuable information and help predict relations. This information may encode entity type, entity compatibility with each others and entity bias to participate in a relation. Given that our entity embeddings are simple GloVe vectors, we believe there is still a large room for improvement.

Sentence Aggregation We compare different ways of aggregating sentences into a single vector including maxpooling (baseline, originally proposed in Jiang et al. (2016)), attention (Lin et al., 2016) and our proposed maxpooled attention.[10]

Maxpooling works better than soft attention because it is better at picking out useful features from multiple sentences, while attention can only weight the whole representation of the sentence. We hypothesize that our proposed maxpooled attention works better than both because it combines the soft attention's ability to learn and use different weights for different sentences, and the maxpool's ability to pick out useful features from multiple sentence. Another advantage of maxpooled attention over attention is that it helps in cases where bag size equals 1 because the softmax typically used in attention results in a weight of 1 for the sentence rendering that weight useless.

Sentence-Level Supervision The last three lines in Table 1 compare different ways for using sentence-level annotations. The line "baseline + maxpooled att." is copied from the previous line and is the basis for the following two lines. In "additional bags," we add the sentence-level annotations as additional bags along with the distantly supervised data. In "sentence loss," we use the method described in Section 3 for integrating sentence-level supervision. The results show that simply adding the sentence-level supervised data to the distantly supervised data as additional bags has little effect on the performance. This is probably because they change the distribution of the training to differ from the test set. However, adding the sentence-level supervision following our proposed multitask learning improves the results considerably because it allows the model to better filter noisy sentences.

[10]Our reimplementation of Lin et al. (2016) attention differs from what was described in the paper. The unnormalized attention weights of Lin et al. (2016) are $o_j = \mathbf{s}_j \times \mathbf{A} \times \mathbf{q}$ where $\mathbf{s}_j$ is the sentence encoding, $\mathbf{A}$ is a diagonal matrix and $\mathbf{q}$ is the query vector. We tried this but found that implementing it as a feedforward layer with output size $= 1$ works better. The results in Table 1 are for the feedforward implementation.

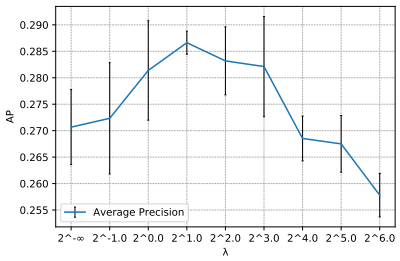

Figure 4: AUC at different λ. X-axis is log-scale.

Selecting Lambda. Although we did not spend much time tuning hyperparameters, we made sure to carefully tune λ (Equation 3) which balances the contribution of the two losses. Early experiments showed that sentence-level loss is typically smaller than bag-level loss, so we experimented with $\lambda \in \{0, 0.5, 1, 2, 4, 8, 16, 32, 64\}$. Figure 4 shows thresholded AUC for different values of λ, where each point is the average of three runs. It is clear that picking the right value for λ has a big impact on the final result.

Qualitative Analysis. An example of a positive bag is shown in Table 2. Our model, which incorporates sentence-level supervision, assigns the most weight to the first sentence while the attention model assigns the most weight to the last sentence (which is less informative for the relation between the two entities). Furthermore, the attention model does not use the other two sentences because their weights are dominated by the weight of the last sentence. We also found that the weights from our model usually range between 0 and 0.08, suggesting the relative values of the weights are informative to the model, even when the absolute values are small.

Attention	This work	Sentences
0.00	0.029	You can line up along the route to cheer for the 32,000 riders, whose 42-mile trip will start in battery park and end with a festival at **Fort Wadsworth** on **Staten Island** .
0.00	0.026	Gateway is a home to the nation's oldest continuing operating lighthouse, Sandy Hook lighthouse, built in 1764; Floyd Bennett field in Brooklyn, which was the city's first municipal airfield; **Fort Wadsworth** on **Staten Island**, which predates the revolutionary war.
0.99	0.027	home energy smart fair, gateway national recreation area, **Fort Wadsworth** visitor center, bay street and school road, **Staten Island**.

Table 2: Weights assigned to sentences by the attention model and our best model. The attention model incorrectly predicts `no_relation`, while our model correctly predicts `neighbourhood_of` for this bag.

5 Related Work

Distant Supervision. The term 'distant supervision' was coined by Mintz et al. (2009) who used relation instances in a KB (Freebase, Bollacker et al., 2008) to identify any sentence in a text corpus where two related entities are mentioned, then developed a classifier to predict the relation. Researchers have since extended this approach for relation extraction (e.g., Takamatsu et al., 2012; Min et al., 2013; Riedel et al., 2013; Koch et al., 2014).

A key source of noise in distant supervision is that sentences may mention two related entities without expressing the relation between them. Hoffmann et al. (2011) used multi-instance learning to address this problem by developing a graphical model for each entity pair which includes a latent variable for each sentence to explicitly indicate the relation expressed by that sentence, if any. Our model can be viewed as an extension of Hoffmann et al. (2011) where the sentence-bound latent variables can also be directly supervised in some of the training examples.

Neural Models for Distant Supervision. More recently, neural models have been effectively used to model textual relations (e.g., Hashimoto et al., 2013; Zeng et al., 2014; Nguyen and Grishman, 2015). Focusing on distantly supervised models, Zeng et al. (2015) proposed a neural implementation of multi-instance learning to leverage multiple sentences which mention an entity pair in distantly supervised relation extraction. However, their model picks only one sentence to represent an entity pair, which wastes the information in the neglected sentences. Jiang et al. (2016) addresses this limitation by maxpooling the vector encodings of all input sentences for a given entity pair. Lin et al. (2016) independently proposed to use attention to address the same limitation. Results in Section 4.2 suggest that maxpooling is more effective than attention for multi-instance learning. Ye et al. (2017) proposed a method for leveraging dependencies between different relations in a pairwise ranking framework.

Sentence-Level Supervision. Despite the substantial amount of work on both fully supervised and distantly supervised relation extraction, the question of how to combine both signals has been mostly ignored in the literature, with a few exceptions. Nguyen and Moschitti (2011) first manually defined a mapping between relation types in YAGO to compatible relation types in ACE 2004 (Doddington et al., 2004), then trained two separate SVM models using the training portion of ACE 2004 and the distantly supervised sentences. Model predictions are then linearly combined to make the final prediction. In contrast, we use a neural model which combines both sources of supervision in a multi-task learning framework (Caruana, 1997). We also do not require a strict mapping between the relation types of the KB and those annotated at the sentence level. Another important distinction is the unit of prediction (at the sentence level vs. at the entity pair level), each of which has important practical applications. Also related is Angeli et al. (2014) who used active learning to improve the multi-instance multi-label model of Surdeanu et al. (2012).

6 Conclusion

We propose two complementary methods to improve performance and reduce noise in distantly supervised relation extraction. The first is incorporating sentence-level supervision and the second is *maxpooled attention*, a novel form of attention. The sentence-level supervision improves sentence encoding and provides supervision for attention weights, while maxpooled attention effectively combines sentence encodings and their weights into a bag encoding. Our experiments show a 10% improvement in AUC (from 0.261 to 0.284) outperforming recently published results on the FB-NYT dataset (Liu et al., 2017).

References

Gabor Angeli, Julie Tibshirani, Jean Wu, and Christopher D Manning. 2014. Combining distant and partial supervision for relation extraction. In *EMNLP*.

Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*.

Razvan C. Bunescu and Raymond J. Mooney. 2005. A shortest path dependency kernel for relation extraction. In *HLT/EMNLP*.

Rich Caruana. 1997. Multitask learning. *Machine Learning*, 28:41–75.

George R. Doddington, Alexis Mitchell, Mark A. Przybocki, Lance A. Ramshaw, Stephanie Strassel, and Ralph M. Weischedel. 2004. The automatic content extraction (ace) program - tasks, data, and evaluation. In *LREC*.

Miao Fan, Deli Zhao, Qiang Zhou, Zhiyuan Liu, Thomas Fang Zheng, and Edward Y. Chang. 2014. Distant supervision for relation extraction with matrix completion. In *ACL*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Kazuma Hashimoto, Makoto Miwa, Yoshimasa Tsuruoka, and Takashi Chikayama. 2013. Simple customization of recursive neural networks for semantic relation classification. In *EMNLP*.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke S. Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *ACL*.

Yi Yao Huang and William Yang Wang. 2017. Deep residual learning for weakly-supervised relation extraction. In *EMNLP*.

Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2017. Distant supervision for relation extraction with sentence-level attention and entity descriptions. In *AAAI*.

Xiaotian Jiang, Quan Wang, Peng Li, and Bin Wang. 2016. Relation extraction with multi-instance multi-label convolutional neural networks. In *COLING*.

Mitchell Koch, John Gilmer, Stephen Soderland, and Daniel S. Weld. 2014. Type-aware distantly supervised relation extraction with linked arguments. In *EMNLP*.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *ACL*.

Tian Yu Liu, Kexiang Wang, Baobao Chang, and Zhifang Sui. 2017. A soft-label method for noise-tolerant distantly supervised relation extraction. In *EMNLP*.

Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. 2013. Distant supervision for relation extraction with an incomplete knowledge base. In *HLT-NAACL*.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL*.

Thien Huu Nguyen and Ralph Grishman. 2015. Relation extraction: Perspective from convolutional neural networks. In *VS@HLT-NAACL*.

Truc-Vien T. Nguyen and Alessandro Moschitti. 2011. Joint distant and direct supervision for relation extraction. In *IJCNLP*.

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

Sebastian Riedel, Limin Yao, and Andrew D McCallum. 2010. Modeling relations and their mentions without labeled text. In *ECML/PKDD*.

Sebastian Riedel, Limin Yao, Andrew D McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *HLT-NAACL*.

Benjamin Roth, Tassilo Barth, Michael Wiegand, and Dietrich Klakow. 2013. A survey of noise reduction methods for distant supervision. In *AKBC@CIKM*.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance multi-label learning for relation extraction. In *EMNLP-CoNLL*.

Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. 2012. Reducing wrong labels in distant supervision for relation extraction. In *ACL*.

Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*.

Hai Ye, Wen-Han Chao, Zhunchen Luo, and Zhoujun Li. 2017. Jointly extracting relations with class ties via effective deep ranking. In *ACL*.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *EMNLP*.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *COLING*.