Semi-Supervised Teacher-Student Architecture for Relation Extraction

Fan Luo, Ajay Nagesh, Rebecca Sharp, and Mihai Surdeanu
University of Arizona, Tucson, AZ, USA
{fanluo, ajaynagesh, bsharp, msurdeanu}@email.arizona.edu

Abstract

Generating a large amount of training data for information extraction (IE) is either costly (if annotations are created manually), or runs the risk of introducing noisy instances (if distant supervision is used). On the other hand, semi-supervised learning (SSL) is a cost-efficient solution to combat lack of training data. In this paper, we adapt Mean Teacher (Tarvainen and Valpola, 2017), a denoising SSL framework to extract semantic relations between pairs of entities. We explore the sweet spot of amount of supervision required for good performance on this binary relation extraction task. Additionally, different syntax representations are incorporated into our models to enhance the learned representation of sentences. We evaluate our approach on the Google-IISc Distant Supervision (GDS) dataset, which removes test data noise present in all previous distance supervision datasets, which makes it a reliable evaluation benchmark (Jat et al., 2017). Our results show that the SSL Mean Teacher approach nears the performance of fully-supervised approaches even with only 10% of the labeled corpus. Further, the syntax-aware model outperforms other syntax-free approaches across all levels of supervision.

1 Introduction

Occurrences of entities in a sentence are often linked through well-defined relations; e.g., occurrences of PERSON and ORGANIZATION in a sentence may be linked through relations such as *employed at*. The task of relation extraction (RE) is to identify such relations automatically (Pawar et al., 2017). RE is not only crucial for populating knowledge bases with triples consisting of the entity pairs and the extracted relations, but it is also important for any NLP task involving text understanding and inference, such as question answering. In this work, we focus on binary RE, such as identifying the *nationality* relation between two entities, *Kian Tajbakhsh* and *Iran*, in the sentence: *Kian Tajbakhsh is a social scientist who lived for many years in England and the United States before returning to Iran a decade ago.*

Semi-supervised learning (SSL) methods have been shown to work for alleviating the lack of training data in information extraction. For example, bootstrapping learns from a few seed examples and iteratively augments the labeled portion of the data during the training process (Yarowsky, 1995; Collins and Singer, 1999; Carlson et al., 2010; Gupta and Manning, 2015, *inter alia*). However, an important drawback of bootstrapping, which is typically iterative, is that, as learning advances, the task often drifts semantically into a related but different space, e.g., from learning women's names into learning flower names (Yangarber, 2003; McIntosh, 2010). Unlike iterative SSL methods such as bootstrapping, SSL teacher-student networks (Tarvainen and Valpola, 2017; Laine and Aila, 2016; Rasmus et al., 2015) make full use of *both* a small set of labeled examples as well as a large number of unlabeled examples in a one-shot, non-iterative learning process. The unsupervised component uses the unlabeled examples to learn a robust representation of the input data, while the supervised component forces these learned representations to stay relevant to the task.

The contributions of our work are:

(1) We provide a novel application of the Mean Teacher (MT) SSL framework to the task of binary relation extraction. Our approach is simple: we build a student classifier using representation learning of the context between the entity mentions that participate in the given relation. When exposed to unlabeled data, the MT framework learns by maximizing the consistency between a student classifier and a teacher that is an average of

past students, where both classifiers are exposed to different noise. For RE, we introduce a strategy to generate noise through word dropout (Iyyer et al., 2015). We provide all code and resources needed to reproduce results[1].

(2) We represent sentences with several types of syntactic abstractions, and employ a simple neural sequence model to embed these representations. We demonstrate in our experiments that the representation that explicitly models syntactic information helps SSL to make the most of limited training data in the RE task.

(3) We evaluate the proposed approach on the Google-IISc Distant Supervision (GDS) dataset introduced in Jat et al. (2017). Our empirical analysis demonstrates that the proposed SSL architecture approaches the performance of state-of-the-art fully-supervised approaches, even when using only 10% of the original labeled corpus.

2 Related work

The exploration of teacher-student models for semi-supervised learning has produced impressive results for image classification (Tarvainen and Valpola, 2017; Laine and Aila, 2016; Rasmus et al., 2015). However, they have not yet been well-studied in the context of natural language processing. Hu et al. (2016) propose a teacher-student model for the task of sentiment classification and named entity recognition, where the teacher is derived from a manually specified set of rule-templates that regularizes a neural student, thereby allowing one to combine neural and symbolic systems. Our MT system is different in that the teacher is a simple running average of the students across different epochs of training, which removes the need of human supervision through rules. More recently, Nagesh and Surdeanu (2018) applied the MT architecture for the task of semi-supervised Named Entity Classification, which is a simpler task compared to our RE task.

Recent works (Liu et al., 2015; Xu et al., 2015; Su et al., 2018) use neural networks to learn syntactic features for relation extraction via traversing the shortest dependency path. Following this trend, we adapt such syntax-based neural models to both of our student and teacher classifiers in the MT architecture.

[1] hidden_for_review

Both Liu et al. (2015) and Su et al. (2018) use neural networks to encode the words and dependencies along the shortest path between the two entities, and Liu et al. additionally encode the dependency subtrees of the words for additional context. We include this representation (words and dependencies) in our experiments. While the inclusion of the subtrees gives Liu et al. a slight performance boost, here we opt to focus only on the varying representations of the dependency path between the entities, without the additional context.

Su et al. (2018) use an LSTM to model the shortest path between the entities, but keep their lexical and syntactic sequences in separate channels (and they have other channels for additional information such as part of speech (POS)). Rather than maintaining distinct channels for the different representations, here we elect to keep both surface and syntactic forms in the same sequence and instead experiment with different degrees of syntactic representation. We also do not include other types of information (e.g., POS) here, as it is beyond the scope of the current work.

There are several more structured (and more complex) models proposed for relation extraction, e.g., tree-based recurrent neural networks (Socher et al., 2010) and tree LSTMs (Tai et al., 2015). Our semi-supervised framework is an orthogonal improvement, and it is flexible enough to potentially incorporate any of these more complex models.

3 Mean Teacher Framework

Our approach repurposes the Mean Teacher framework for relation extraction. This section an overview of the general MT framework. The next section discusses the adaptation of this framework for RE.

The Mean Teacher (MT) framework, like other teacher-student algorithms, learns from a limited set of labeled data coupled with a much larger corpus of unlabeled data. Intuitively, the larger, unlabeled corpus allows the model to learn a *robust representation* of the input (i.e., one which is not sensitive to noise), and the smaller set of labeled data constrains this learned representation to also be *task-relevant*. This is similar in spirit to an auto-encoder, which learns a representation that is robust to noise in the input. However, auto-encoders generally suffer from a confirmation bias, especially when used in a semi-supervised setting (i.e.,

they tend to overly rely on their own predictions instead of the gold labels (Tarvainen and Valpola, 2017; Laine and Aila, 2016)), providing the motivation for MT. Specifically, to mitigate confirmation bias, and regularize the model, the teacher weights in MT are not directly learned, but rather they are an *average* of the learned student weights, similar in spirit to the averaged perceptron. This provides a more robust scaffolding that the student model can rely on during training when gold labels are unavailable (Nagesh and Surdeanu, 2018).

The MT architecture is summarized in Figure 1. It consists of two models, *teacher* and *student*, both with identical architectures (but different weights). While the weights for the student model are learned through standard backpropagation, the weights in the teacher network are set through an exponential moving average of the student weights. The same data point, augmented with noise (see Section 4.2), is input to both the teacher and the student models.

The MT algorithm is designed for semi-supervised tasks, where only a few of the data points are labeled in training. The cost function is a linear combination of two different type of costs: *classification* and *consistency*. The classification cost applies to labeled data points, and can be instantiated with any standard supervised cost function (e.g., we used categorical cross-entropy, which is based on the softmax over the label categories). The consistency cost is used for unlabeled data points, and aims to minimize the differences in predictions between the teacher and the student models. The consistency cost, J is:

$$J(\theta) = \mathrm{E}_{x,\eta',\eta}\left[\|f(x,\theta',\eta') - f(x,\theta,\eta)\|^2\right] \quad (1)$$

where, given an input x, this function is defined as the expected distance between the prediction of the student model ($f(x,\theta,\eta)$, with weights θ and noise η) and the prediction of the teacher model ($f(x,\theta',\eta')$ with weights θ' and noise η'). Here, our predictions are the models' output distributions (with a softmax) across all labels.

Importantly, as hinted above, only the student model is updated via backpropagation. On the other hand, the gradients are not backpropagated through the teacher weights, rather they are deterministically updated after each mini-batch of gradient descent in the student. The update uses an exponentially-weighted moving average (EMA) that combines the previous teacher with the latest version of the student:

$$\theta'_t = \alpha\theta'_{t-1} + (1-\alpha)\theta_t \quad (2)$$

where θ'_t is defined at training step t as the EMA of successive weights θ. This averaging strategy is reminiscent of the average perceptron, except in this case the average is not constructed through an error-driven algorithm, but, instead, it is controlled by a hyper parameter α.

4 Task-specific Representation

The MT framework contains two components that are task specific. The first is the representation of the input to be modeled, which consists of entity mention pairs and the context between the two entity mentions. The second is the strategy for inserting noise in the inputs received by the student and teacher models. We detail both these components here.

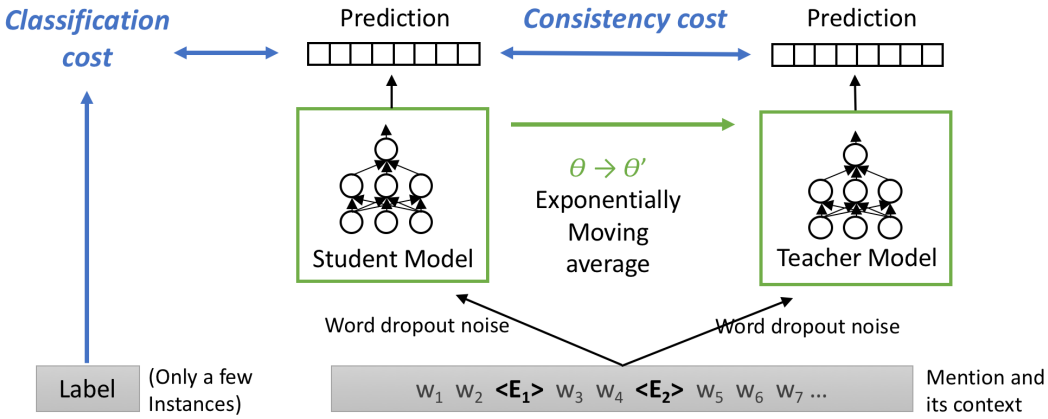
4.1 Input Representations

Within the Mean Teacher framework, we investigate input representations of four types of syntactic abstraction. Each corresponds to one of the inputs shown in Figure 2 for the example (*Robert Kingsley*, `perGraduatedInstitution`, *University of Minnesota*).

Average: Our shallowest learned representation uses the surface form of the entities (e.g., *Robert Kingsley* and *University of Minnesota*) and the context between them. If there are several mentions of one or both of the entities in a given sentence, we used the closest pair. The model averages the word embeddings for all tokens and then passes it through a fully-connected feed-forward neural network, with one hidden layer and finally to an output layer for classification into the the relation labels.

Surface LSTM (surfaceLSTM): The next learned representation also uses the surface form of the input, i.e., the sequence of words between the two entities, but replaces the word embedding average with a single-layer bidirectional LSTM, which have been demonstrated to encode some syntactic information (Peters et al., 2018). The representation from the LSTM is passed to the feed-forward network as above.

Head LSTM (headLSTM): Under the intuition that the trigger is the most important lexical item in the context of a relation, relation extraction



Example: (Robert_Kinglsey, `perGraduatedInstitution`, University_of_Minnesota)

Sentence: **<Robert_Kingsley>** (1903-1988) was an American legal scholar and California judge who graduated from the **<University_of_Minnesota>**.

Figure 1: The Mean Teacher (MT) framework for relation extraction which makes use of a large, unlabeled corpus and a small number of labeled data points. Intuitively, the larger, unlabeled corpus allows the model to learn a *robust representation* of the input (i.e., one which is not sensitive to noise), and the smaller set of labeled data constrains this learned representation to also be *task-relevant*. Here, for illustration, we depict the training step with one labeled example. The MT framework consists of two models: a *student model* which is trained through backpropagation, and a *teacher model* which is not directly trained, but rather is an average of previous student models (i.e., its weights are updated through an exponential moving average of the student weights at each epoch). Each model takes the same input, but augmented with different noise (here, word dropout). The inputs consist of (a) a pair of entity mentions (here, $\langle E_1 \rangle$*: Robert_Kingsley* and $\langle E_2 \rangle$*: University_of_Minnesota*), and (b) the context in which they occur (i.e., *(1903-1988) was an American legal scholar and California judge who graduated from the*). Each model produces a prediction for the label of the data point (i.e., a distribution over the classes). There are two distinct costs in the MT framework: the *consistency cost* and the *classification cost*. The consistency cost constrains the output label distribution of the two models to be the similar, and it is applied to both labeled and unlabeled inputs (as these distributions can be constrained even if the true label is unknown) to ensure that the learned representations (distributions) are not sensitive to the applied noise. When the model is given a labeled input, it additionally assigns a classification cost using the prediction of the student model. This guides the learned representation to be useful to the task at hand.

can often be distilled into the task of identifying and classifying *triggers*, i.e., words which signal specific relations (Yu and Ji, 2016). For example, the verb *died* is highly indicative of the `placeOfDeath` relation. There are several ways of finding a candidate trigger such as the PageRank-inspired approach of Yu and Ji (2016). Here, for simplicity, we use the governing head of the two entities within their token interval (i.e., the node in the dependency graph that dominates the two entities) as a proxy for the trigger. We then create the shortest dependency paths from it to each of the two entities, and concatenate these two sequences to form the representation of the corresponding relation mention. This is shown as the head-lexicalized syntactic path in Figure 2. For this representation, we once again pass this token sequence to the LSTM and subsequent feed-forward network.

Syntactic LSTM (synLSTM): Compared with headLSTM, our syntactic LSTM traverses the directed shortest path between the two entities through the dependency syntax graph for the sentence, instead of concatenated shortest dependency paths between the trigger and each entity. As shown in the fully lexicalized syntactic path in Figure 2, we include in the representation the directed dependencies traversed (e.g., `<nsubj` for an incoming `nsubj` dependency), as well as all the words along the path (i.e., *scholar* and *graduated*)[2]. We feed these tokens, both words and dependencies, to the same LSTM as above.

4.2 Noise Regularization

One critical component of the Mean Teacher framework is the addition of independent noise to both the student and the teacher models (Laine

[2]For multi-sentence instances where no single sentence contains both entities, in order to obtain a syntactic parse, we concatenate the closest sentences with each of the entities with a semi-colon and use that as our sentence. While this could be resolved more cleanly with discourse information or cross-sentence dependencies, that is beyond the scope of the current work.

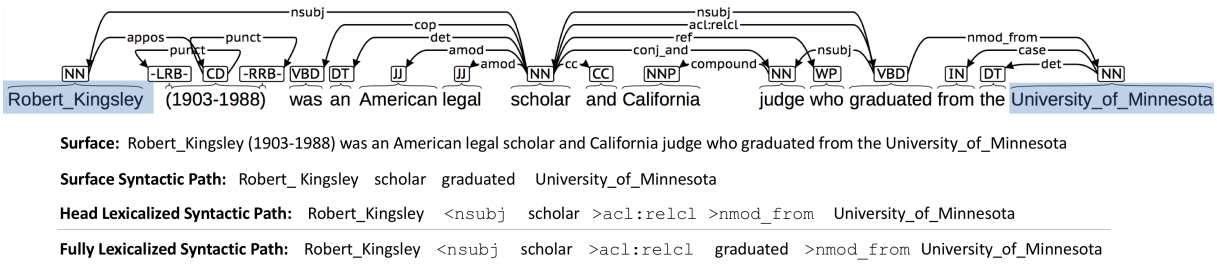


Figure 2: Examples of the varying degrees of syntactic structure used for our models. The *surface* representation consists of a bag of entities (shown in blue) and the words that come between them. The *surface syntactic path* includes the words along the shortest path between the entities. The *head lexicalized syntactic* representation contains the governing head of the structure that dominates the two entities, and the directed dependencies connecting it to each of the entities. The *fully lexicalized syntactic* representation contains the directed dependencies along the shortest path in addition to the words along the shortest path. For both *head lexicalized syntactic* and *fully lexicalized syntactic* representation, the dependency notations include the dependency labels as well as the direction of the dependency arc (<: right-to-left, >: left-to-right). Both words and syntactic dependencies are converted to embeddings that are trained along with the rest of the model.

and Aila, 2016; Tarvainen and Valpola, 2017). While in some tasks, e.g., image classification, Gaussian noise is added to the network (Rasmus et al., 2015), here we opted to follow the example of Iyyer et al. (2015) and Nagesh and Surdeanu (2018) and use word dropout for the needed noise. For each training instance we randomly drop k tokens from the input sequence.[3] The random dropout for the student model is independent from that of the teacher model, in principle forcing the learned representations to be robust to noise and learn noise-invariant abstract latent representations for the task.

5 Experiments

5.1 Data

We evaluate our approach on **Google-IISc Distant Supervision (GDS) dataset**. This is a distant supervision dataset introduced by Jat et al. (2017) that aligns the Google Relation Extraction Corpus[4] with texts retrieved through web search. Importantly, the dataset was curated to ensure that at least one sentence for a given entity pair supports the corresponding relation between the entities. It contains five relations including *NA*. The training partition contains 11,297 instances (2,772 are `NA`), the development partition contains 1,864 instances (447 `NA`), and the test partition contains 5663 instances (1360 `NA`). The dataset is divided such that there is no overlap among entity pairs between these partitions.

5.2 Semi-Supervised Setup

In order to examine the utility of our semi-supervised approach, we evaluate our approach with different levels of supervision by artificially manipulating the fully-labeled datasets described in Section 5.1 to contain varying amounts of unlabeled instances. To do this, we incrementally selected random portions of the training data to serve as labeled data, the rest being treated as unlabeled data (i.e., their labels are masked to not be visible to the classifier). In other words, this unlabeled data was used to calculate the *consistency cost*, but not the *classification cost*. We experimented with using 1%, 10%, 50%, and 100% of the labeled training data for supervision.

5.3 Baselines

Previous work: We compare our model against the previous state-of-the-art work in Jat et al. (2017). They propose two *fully-supervised* models: a bidirectional gated recurrent unit neural network with word attention (their BGWA model) and a piecewise convolutional neural network (PCNN) with an entity attention layer (their EA model), which is itself based on the PCNN approach of Zeng et al. (2015). Since we use single models only, we omit Jat et al.'s supervised ensemble for a more direct comparison. Note these two approaches are state-of-the-art methods for RE that rely on more complex attention-based models. On the other hand, in this work we use only vanilla LSTMs for both of our student and teacher models. However, since the MT framework is agnostic to the student model, in fu-

[3] In this work we use $k = 1$. In initial experiments, other values of k did not change results significantly.

[4] https://research.googleblog.com/2013/04/50000-lessons-on-how-to-read-relation.html

ture work we can potentially further increase performance by incorporating these more complex methods in our semi-supervised approach.

Student Only: In order to determine the contribution of the Mean Teacher framework itself, in each setting (amount of supervision and input representation) we additionally compare against a baseline consisting of our student model trained without the teacher (i.e., we remove the consistency cost component from the cost function).

5.4 Model Tuning

We lightly tuned the approach and hyperparameters on the development partitions. We built our architecture in PyTorch[5], a popular deep learning library, extending the framework of Tarvainen and Valpola (2017)[6]. For our model, we used a bidirectional LSTM with a hidden size of 50. The subsequent fully connected network has one hidden layer of 100 dimensions and *ReLU* activations. The training was done with the Adam optimizer (Kingma and Ba, 2015) with the default learning rate (0.1). We initialized our word embeddings with GloVe 100-dimensional embeddings (Pennington et al., 2014)[7]. The dependency embeddings were also of size 100 and were randomly initialized. Both of these embedding were allowed to update during training. Any word or token which occurred fewer than 5 times in a dataset was treated as unknown. For our word dropout, we removed one word from each input at random. MT has some additional parameters: the *consistency cost weight* (the weight given to the consistency component of the cost function) and the *EMA decay rate* (α in Eq. 2). We set these to be 1.0 and 0.99 respectively. Akin to a burn-in at the beginning of training, we had an initial consistency cost weight of 0.0 and allowed it to ramp up to the full value over the coarse of training using a *consistency ramp up* of 5.

During training, we saved model checkpoints every 10 epochs, and ran our models up to a maximum of 200 epochs. We chose the best model by tracking the teacher performance on the development partition and using the model checkpoint immediately following the maximum performance.

[5] https://pytorch.org

[6] https://github.com/CuriousAI/mean-teacher

[7] https://nlp.stanford.edu/projects/glove

6 Results

We evaluate our approach on the GDS dataset, across the various levels of supervision detailed in Section 5. The precision, recall, and F1 scores are provided in Table 1.

In the evaluation, we consider only the top label assigned by the model, counting it as a match only if (a) it matches the correct label and (b) the label is not `NA`. For each setting, we provide the final performance of both the student network and the teacher network. We also compare against the baseline student network (i.e., the network trained without the consistency cost).

In Table 1 we see that in general the MT framework improves performance over the student-only baseline, with the sole exception of when there is only 1% supervision. The large performance gap between the 1% and 10% supervision configurations for the student-only model indicates that 1% supervision provides an inadequate amount of labeled data to support learning the different relation types. Further, when the unlabeled data overwhelms the supervised data, the MT framework encourages the student model to drift away from the correct labels through the consistence cost. Therefore, in this situation with insufficient supervision, the student-only approach performs better than MT. This result highlights the importance of a balance between the supervised and unsupervised training partitions within the MT framework.

Among our MT models, we see that while the surfaceLSTM models perform better than the Average models, the models which explicitly represent the syntactic dependency path between the entities have the strongest performance. This is true for both the student-only as well as the MT framework. In particular, we find that the best performing model is the synLSTM. These results show that: (a) while surface LSTMs may loosely model syntax (Linzen et al., 2016), they are still largely complemented by syntax-based representation, and, more importantly, (b) syntax provides improved generalization power over surface information.

6.1 Comparison to Prior Work

We additionally compare our Mean Teacher synLSTM model, against the BGWA and EA models of Jat et al. (2017). For an accurate comparison, here we follow their setup closely. For each entity pair that occurs in test, we aggregate

Model		100% Prec / Recall / F1	50% Prec / Recall / F1	10% Prec / Recall / F1	1% Prec / Recall / F1
Student-Only Baseline					
Average	Student	0.724 / 0.793 / 0.757±0.00	0.716 / 0.774 / 0.744±0.01	0.718 / 0.684 / 0.700±0.01	0.624 / 0.572 / 0.597±0.01
surfaceLSTM	Student	0.754 / 0.830 / 0.790±0.01	0.748 / 0.834 / 0.788±0.01	0.735 / 0.752 / 0.744±0.00	0.691 / 0.622 / 0.655±0.01
headLSTM	Student	0.739 / 0.808 / 0.772±0.01	0.725 / 0.816 / 0.767±0.00	0.697 / 0.723 / 0.709±0.01	0.633 / 0.630 / 0.631±0.01
synLSTM	Student	0.757 / 0.828 / 0.791±0.00	0.759 / 0.827 / 0.791±0.00	0.716 / 0.770 / 0.742±0.01	0.667 / 0.677 / **0.671±0.01**
Mean Teacher Framework					
Average	Student	0.719 / 0.717 / 0.716±0.04	0.739 / 0.718 / 0.728±0.01	0.715 / 0.645 / 0.677±0.02	0.611 / 0.509 / 0.555±0.02
	Teacher	0.733 / 0.767 / 0.750±0.00	0.724 / 0.747 / 0.735±0.00	0.718 / 0.649 / 0.682±0.01	0.611 / 0.503 / 0.552±0.01
surfaceLSTM	Student	0.760 / 0.745 / 0.752±0.01	0.762 / 0.792 / 0.777±0.00	0.725 / 0.712 / 0.718±0.00	0.546 / 0.440 / 0.486±0.07
	Teacher	0.761 / 0.819 / 0.789±0.00	0.760 / 0.817 / 0.787±0.00	0.735 / 0.733 / 0.734±0.01	0.538 / 0.438 / 0.482±0.06
headLSTM	Student	0.718 / 0.766 / 0.741±0.01	0.728 / 0.786 / 0.756±0.01	0.700 / 0.684 / 0.692±0.01	0.613 / 0.525 / 0.565±0.00
	Teacher	0.728 / 0.815 / 0.769±0.00	0.731 / 0.800 / 0.764±0.00	0.703 / 0.717 / 0.710±0.01	0.611 / 0.527 / 0.566±0.01
synLSTM	Student	0.753 / 0.780 / 0.766±0.01	0.751 / 0.826 / 0.786±0.01	0.724 / 0.732 / 0.728±0.01	0.634 / 0.590 / 0.609±0.03
	Teacher	0.764 / 0.846 / **0.803±0.00**	0.763 / 0.833 / **0.796±0.00**	0.734 / 0.759 / **0.746±0.00**	0.632 / 0.574 / 0.601±0.03

Table 1: Performance on the GDS dataset for our baseline models (i.e, student-only without the Mean Teacher framework) and for our Mean Teacher models (both the student and teacher performance is provided). We report precision, recall, and F1 score for each of the input representations (Average, surfaceLSTM, headLSTM, and synLSTM), with varying proportions of training labels (100%, 50%, 10% and 1% of the training data labeled). Each value is the average of three runs with different random seed initializations, and for the F1 scores we additionally provide the standard deviation across the different runs.

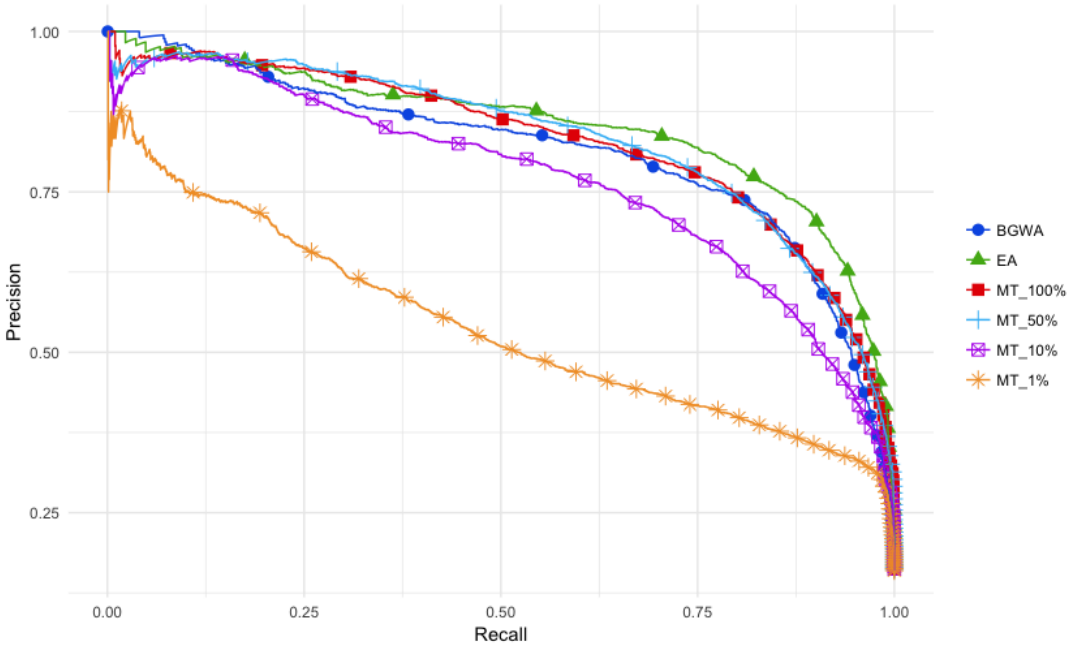


Figure 3: Precision-recall curves for the GDS dataset. We include two prior systems (BGWA and EA) as well as our Mean Teacher approach using our best performing input representation (synLSTM) for increasing amounts of supervision (from 1% of the training labels to 100%, or fully supervised).

the model's predictions for all mentions of that entity pair using the Noisy-Or formula into a single distribution for our model across *all* labels, except for NA, and plot the precision-recall (PR) curve accordingly.

Data for the BGWA and EA models was taken from author-provided results files[8], allowing us to plot additional values beyond what was previously published. The PR curves for each level of supervision are shown in Figures 3.

We observe that our fully supervised model performs very similarly to the fully-supervised model of Jat et al. (2017). This is encouraging, considering that our approach is simpler, e.g., we do not have an attention model. Critically, we see that under the Mean Teacher framework, this pattern continues even as we decrease the amount of supervision such that even with only 10% of the original training data we approach their fully-supervised performance. It is not until we use two orders of magnitude fewer labeled examples that we see a major degradation in performance. This demonstrates the utility of our syntax-aware MT approach for learning robust representations for relation extraction.

7 Conclusion

This paper introduced a neural model for semi-supervised relation extraction. Our syntactically-informed, semi-supervised approach learns effectively to extract a number of relations from very limited labeled training data by employing a Mean Teacher (MT) architecture. This framework combines a student trained through backpropagation with a teacher that is an average of past students. Each model is exposed to different noise, which we created in this work through word dropout. The approach uses unlabeled data through a consistency cost, which encourages the student to stay close to the predictions of the teacher, and labeled data, through a standard classification cost. The consistency requirement between the student and the teacher ensures that the learned representation of the input is robust to noise, while the classification requirement ensures that this learned representation encodes the task-specific information necessary to correctly extract relations between entities.

We empirically demonstrated that our approach is able to perform close to more complex, fully-supervised approaches using only 10% of the training data for relation extraction. This work further supports our previous work on MT for the task of named entity classification, where we observed similar gains (Nagesh and Surdeanu, 2018). Further, we show that the MT framework performs reliably for various input representations (from purely surface forms all the way to primarily syntactic). We additionally demonstrate that explicitly representing syntax, even in simplified ways, is beneficial to the models across all levels of supervision.

[8] https://github.com/SharmisthaJat/RE-DS-Word-Attention-Models

8 Acknowledgments

This work was supported by the Defense Advanced Research Projects Agency (DARPA) under the World Modelers program, grant number W911NF1810014, and by the Bill and Melinda Gates Foundation HBGDki Initiative. Mihai Surdeanu declares a financial interest in lum.ai. This interest has been properly disclosed to the University of Arizona Institutional Review Committee and is managed in accordance with its conflict of interest policies.

References

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proc. of AAAI*.

Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In *Proc. of EMNLP*.

Sonal Gupta and Christopher D. Manning. 2015. Distributed representations of words to guide bootstrapped entity classifiers. In *Proc. of NAACL*.

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. 2016. Harnessing deep neural networks with logic rules. *arXiv preprint arXiv:1603.06318*.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proc. of ACL-IJCNLP*, volume 1, pages 1681–1691.

Sharmistha Jat, Siddhesh Khandelwal, and Partha Talukdar. 2017. Improving distantly supervised relation extraction using word and entity based attention. In *Proc. of AKBC*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Samuli Laine and Timo Aila. 2016. Temporal ensembling for semi-supervised learning. *CoRR*, abs/1610.02242.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of lstms to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Yang Liu, Furu Wei, Sujian Li, Heng Ji, Ming Zhou, and Houfeng WANG. 2015. A dependency-based neural network for relation classification. In *Proc. of ACL-IJCNLP*, pages 285–290, Beijing, China.

Tara McIntosh. 2010. Unsupervised discovery of negative categories in lexicon bootstrapping. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 356–365.

Ajay Nagesh and Mihai Surdeanu. 2018. An exploration of three lightly-supervised representation learning approaches for named entity classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2312–2324.

Sachin Pawar, Girish K. Palshikar, and Pushpak Bhattacharyya. 2017. Relation extraction : A survey.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*, pages 2227–2237.

Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. 2015. Semi-supervised learning with ladder networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Proc. of NIPS*, pages 3546–3554. Curran Associates, Inc.

Richard Socher, Christopher D Manning, and Andrew Y Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, volume 2010, pages 1–9.

Yu Su, Honglei Liu, Semih Yavuz, Izzeddin Gur, Huan Sun, and Xifeng Yan. 2018. Global relation embedding for relation extraction. In *Proc. of NAACL-HLT*, pages 820–830, New Orleans, Louisiana.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.

Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, pages 1195–1204.

Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. 2015. Classifying relations via long short term memory networks along shortest dependency paths. In *Proc. of EMNLP*, pages 1785–1794.

Roman Yangarber. 2003. Counter-training in discovery of semantic patterns. In *Proc. of ACL*.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proc. of ACL*.

Dian Yu and Heng Ji. 2016. Unsupervised person slot filling based on graph mining. In *Proc. of ACL*, volume 1, pages 44–53.

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