An Exploration of Three Lightly-supervised Representation Learning Approaches for Named Entity Classification

Ajay Nagesh and Mihai Surdeanu
University of Arizona, Tucson, AZ, USA
`{ajaynagesh, msurdeanu}@email.arizona.edu`

Abstract

Several semi-supervised representation learning methods have been proposed recently that mitigate the drawbacks of traditional bootstrapping: they reduce the amount of semantic drift introduced by iterative approaches through one-shot learning; others address the sparsity of data through the learning of custom, dense representation for the information modeled. In this work, we are the first to adapt three of these methods, most of which have been originally proposed for image processing, to an information extraction task, specifically, named entity classification. Further, we perform a rigorous comparative analysis on two distinct datasets. Our analysis yields several important observations. First, all representation learning methods outperform state-of-the-art semi-supervised methods that do not rely on representation learning. To the best of our knowledge, we report the latest state-of-the-art results on the semi-supervised named entity classification task. Second, one-shot learning methods clearly outperform iterative representation learning approaches. Lastly, one of the best performers relies on the mean teacher framework (Tarvainen and Valpola, 2017), a simple teacher/student approach that is independent of the underlying task-specific model.

1 Introduction

The paucity of labeled datasets presents tremendous challenges to training machine learning (ML) systems in the real world. Most natural language processing (NLP) tasks are data hungry. The recent success of deep learning for NLP tasks is at least partially due to the fact that such models are exposed to large quantities of labeled data. However, obtaining such rich labeled data is a costly proposition and seldom feasible in many realistic scenarios.

Semi-supervised learning (SSL) mitigates the supervision cost by combining a minimal amount of labeled data with a large amount of unlabeled data. In information extraction (IE), SSL often takes the form of bootstrapping, which starts with a few seed examples, e.g., "Barack Obama" as an example of a person's name, and continues with an iterative approach that alternates between learning extraction patterns such as word n-grams, e.g., the pattern "`@ENTITY , former president`" could be used to extract person names, and applying these patterns to extract the desired structures (entities, relations, etc.) (Riloff, 1996; Abney, 2007; Carlson et al., 2010b; Gupta and Manning, 2014; Gupta and Manning, 2015, inter alia). There are, however, two drawbacks to this direction. First, as learning in this iterative framework advances, the task often *drifts semantically* (Komachi et al., 2008) into a related but different space, e.g., from learning women names into learning flower names (McIntosh, 2010; Yangarber, 2003). Second, the statistics used to estimate the quality of the learned patterns are often *brittle* due to the sparsity of the extraction patterns.

Recently, several semi-supervised representation learning methods have been proposed to mitigate these issues. One direction learns custom representations for both patterns and the structures to be extracted to mitigate the sparsity mentioned above (see Section 4). A different direction, illustrated by the ladder networks (LN) and mean teacher (MT) frameworks (Rasmus et al., 2015; Tarvainen and Valpola, 2017), replaces the iterative bootstrapping with one-shot learning driven by teacher/student architectures, trained to maximize consistency of predictions between the teacher and the student on the unlabeled data (Sections 5 and 6). Despite this progress, several important questions remain unanswered. First, most of these methods have been proposed for vision, and it is unclear how useful they are for lightly-supervised natural language processing tasks. Second, it is unknown how well these methods compare against "traditional" bootstrapping that does not use representation learning. Third, it is unclear how these representation learning methods compare against each other.

In this work, we answer the above questions by adapting all these methods to the same IE task, and performing a rigorous comparative analysis. Specifically, the contributions of this paper are the following:

- We are the first to adapt the mean teacher (MT) framework (Tarvainen and Valpola, 2017) to a semi-supervised IE task, in particular named entity classification (NEC).
- We implement and evaluate the three semi-supervised representation learning approaches summarized above for the same NEC task, using two distinct datasets (Pradhan et al., 2013; Tjong Kim Sang and De Meulder, 2003), the same inputs, and the same evaluation measures to understand how they compare both against each other and against more traditional semi-supervised NLP approaches.
- We perform a thorough comparative analysis of all these methods, which highlights several important observations. First, our analysis confirms that all these representation learning methods outperform state-of-the-art semi-supervised methods that do not rely on representation learning (Gupta and Manning, 2015; Gupta and Manning, 2014; Zhu and Ghahramani, 2002). Second, one-shot learning methods (the teacher/student approaches here) have superior performance compared to iterative representation learning approaches that are driven by objective functions tailored for the task at hand (e.g., which maximize the similarity of embeddings for entities in the same class). Lastly, the teacher/student approaches perform comparatively – each outperforms the other on one dataset – despite the fact that MT is a simpler framework than LN. That is, in the mean teacher framework both student and teacher are decoupled whereas in ladder networks there are connections between the two, which makes porting LNs to new tasks more difficult.

2 Related Work

There is a large body of work in semi-supervised learning for NLP, which encompasses many different types of techniques such as self-training or bootstrapping (Riloff, 1996; Abney, 2007; Carlson et al., 2010a; Carlson et al., 2010b; McIntosh, 2010; Gupta and Manning, 2015, inter alia), co-training (Blum and Mitchell, 1998), or graph-based methods such as label propagation (Delalleau et al., 2005). Among these, perhaps the most-widely adopted approach in the NLP field is bootstrapping, which has been used in many applications, including information extraction (Carlson et al., 2010a; Gupta and Manning, 2014; Gupta and Manning, 2015), lexicon acquisition (Neelakantan and Collins, 2015), named entity classification (Collins and Singer, 1999) and sentiment analysis (Rao and Ravichandran, 2009). However, as mentioned, most of these approaches suffer from brittle statistics due to the sparsity of the patterns learned, and from semantic drift, due to their iterative nature. The methods we investigate in this paper address these two limitations through representation learning and one-shot learning.

Auto-encoder frameworks have received considerable attention in the machine learning community recently. Such frameworks include recursive auto-encoders (Socher et al., 2011), denoising auto-encoders (Vincent et al., 2008), and others. They are primarily used as a pre-training mechanism before supervised training. Recently, such networks have also been used for semi-supervised learning as they are more amenable to combining supervised and unsupervised components of the objective functions (Zhai and Zhang, 2015).

Ladder networks (LN) are stacked denoising auto-encoders with skip-connections in the intermediate layers (Rasmus et al., 2015; Valpola, 2014). LNs have been shown to produce state-of-the-art performance on both supervised and semi-supervised tasks on the MNIST dataset in image processing. Our work is among the first to apply LNs to NLP. While similar in spirit to Zhang et al. (2017), the only other work we found that applies a denoising auto-encoder to a semi-supervised spelling correction task, our work is much simpler, since it uses a multi-layer perceptron instead of convolution-deconvolution operations (see Section 5). Further, we demonstrate that LNs perform very well on a complex IE task, considerably outperforming several state-of-the-art approaches. In the same space, teacher-student networks have shown to provide state-of-the-art semi-supervised learning results on several image processing tasks. The most prominent example of such a network is the Mean Teacher framework (Tarvainen and Valpola, 2017).To our knowledge, we are the first to apply the MT framework to an IE task (Section 6).

Distributed representations of words serve as underlying representation for many NLP tasks. For example, Levy and Goldberg (2014a) generalized the `word2vec` algorithm (Mikolov et al., 2013) to use any arbitrary context instead of just individual words. Faruqui et al. (2015) demonstrated that embeddings learned without supervision can be retro-fitted to better conform to some semantic lexicon. Generating custom embeddings that encode more specific semantic properties has been shown to be useful for downstream tasks (Sharp et al., 2016). Riedel et al. (2013; Toutanova et al. (2015; Toutanova et al. (2016) used knowledge bases in conjunction with surface patterns to learn custom representations for relation extraction. Note, however, that most of these works that customize embeddings for a specific task rely on some form of supervision. In contrast, our approach that learns a custom representation for the NEC task is lightly supervised, with a only few seed examples per category (Section 4).

3 Task and Approaches

Named entity classification

We implement and evaluate the proposed semi-supervised learning approaches on the task of named entity classification (NEC), defined as identifying the correct label of an entity mention in a given context. In our setting, the context of a mention is defined as all the patterns that match that specific mention in its context. In this work, we consider patterns to be n-grams of size up to 4 tokens on either side of an entity. For instance, "`@ENTITY , former President`" is one of the patterns learned for the class `person`. Using all these patterns as input, the classifier must infer the mention's correct label. Importantly, the NEC task can be defined at mention level, i.e., as the classification of *individual mentions* of named entities as defined above, or at entity level, i.e., as the identification of all labels that apply to *all mentions* of a given entity jointly (e.g., "Washington" = {`person`, `location`}). Here we focus on mention classification, although in some of our evaluations we revert to entity classification, to be able to compare against other approaches.

Semi-supervised representation learning

We propose three representation learning approaches for the NEC task: (i) Embedding-based bootstrapping (Emboot), (ii) ladder networks (LN), and (iii) mean teacher (MT) architectures.

Emboot (Valenzuela-Escárcega et al., 2018) is an adaptation of iterative bootstrapping-based approaches, coupled with a component that learns custom representations (embeddings) for both the entities and the patterns learned. As mentioned, such iterative approaches suffer from semantic drift, i.e., as the learning advances, the learning often drifts into a related but different space. LN and MT are based on the emerging paradigm of teacher-student architectures: both approaches aim to reconcile differences between the teacher and the student models by minimizing a consistency cost when their predictions do not concur. Teacher-student architectures are more robust to semantic drift due to their single-shot semi-supervised learning algorithms.

Note that, of the above architectures, Emboot and MT additionally learn custom representations for the unsupervised data provided which could potentially be informative representations to be used in a downstream component aimed at interpreting the classifier decisions. Further, MT is the most flexible/customizable of all these architectures, as it is largely independent of the underlying task-specific model. We show in our experiments that this simplicity is coupled with good performance on two different datasets.

In the next three sections, we discuss each of these approaches in detail.

4 Embedding-based Bootstrapping (Emboot)

This algorithm (based on our earlier work (Valenzuela-Escárcega et al., 2018)) iteratively grows a pool of known entities (entPool$_c$) and patterns (patPool$_c$) for each category of interest c, and learns custom embeddings for both. These pools are initialized with a few seed examples (seeds$_c$) for each category. For example, in our experiments we initialize the pool for a `person names` category with 10 names such as *Bill Clinton* and *Mother Teresa*. Then the algorithm iteratively applies the following three steps for a specified number of epochs:

(1) *Learning custom embeddings*: The algorithm learns custom embeddings for all entities and patterns observed in the dataset, using the current entPool$_c$s. We train our embeddings for both entities and patterns by maximizing the objective function J:

$$J = \text{Skip-gram} + \text{Attract} + \text{Repel} \tag{1}$$

where the individual components of the objective function designed to model both the unsupervised, language model part of the task as well as the light supervision coming from the seed examples. The *Skip-gram* term is similar to Eq. 4 of Mikolov et al. (2013) (skip-gram with negative sampling), but, crucially, adapted to operate over *entities* (rather than words), and a context consisting of the bag of all *patterns* that match each entity (rather than context words). The *Attract* term, encourage entities or patterns in the same pool to be close to each other (*E.g.* two person entities should be attracted to each other), whereas the *Repel* term encourages that the pools be mutually exclusive, which is a soft version of the counter training approach of Yangarber (2003) or the weighted mutual-exclusive bootstrapping algorithm of McIntosh and Curran (2008) (e.g., person names should be far from organization names in the semantic embedding space). They are depicted by the following two equations:

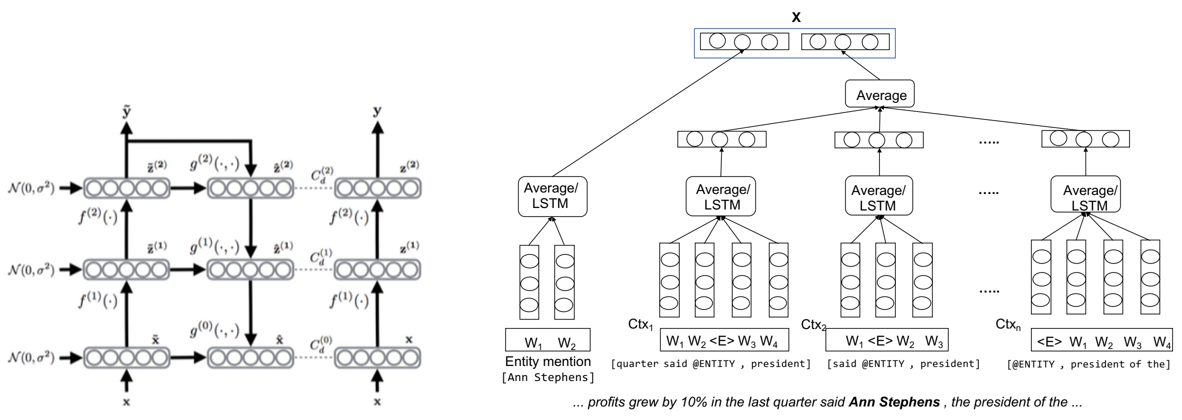
$$\text{Attract} = \sum_{P} \sum_{x1,x2 \in P} \log(\sigma(V_{x1}^{\top} V_{x2})) \tag{2}$$

$$\text{Repel} = \sum_{\substack{P1,P2 \\ \text{if } P1 \neq P2}} \sum_{x1 \in P1} \sum_{x2 \in P2} \log(\sigma(-V_{x1}^{\top} V_{x2})) \tag{3}$$

where P is the entity/pattern pool for a category, $x1, x2$ are entities/patterns in said pool in Eq 2. and $P1, P2$ are different pools, and $x1$ and $x2$ are entities/patterns in $P1$, and $P2$, respectively.

(2) *Pattern promotion*: We generate the patterns that match the entities in each pool entPool$_c$, rank those patterns using point-wise mutual information (PMI) with the corresponding category, and select the top ranked patterns for promotion to the corresponding pattern pool patPool$_c$.

(3) *Entity promotion*: Entities are promoted to entPool$_c$ using a classifier that estimates the likelihood of an entity belonging to each class (Gupta and Manning, 2015). We differ from Gupta and Manning (2015) in that we use a multi-class classifier instead of many binary classifiers, and in the features used. Our feature set includes, for each category c: (a) edit distance between the candidate entity e and current e_cs $\in$ entPool$_c$, (b) the PMI (with c) of the patterns in patPool$_c$ that matched e in the training documents, and (c) similarity between e and e_cs in some semantic space. For the latter feature group, we use two sets of vector representations for entities. The first is the set of custom embedding vectors learned in step (1) for entities. The second includes pre-trained word embeddings (for multi-word entities, we simply average the embeddings of the component words). We use these vectors to compute the cosine similarity score of a given candidate entity e to the entities in entPool$_c$, and add the average and maximum similarities as features. The top 10 entities classified with the highest confidence for each class are promoted to the corresponding entPool$_c$ after each epoch.



x
Average
Average/ LSTM
Average/ LSTM
Average/ LSTM
Average/ LSTM
Ctx1
Ctx2
Ctxn
W1 W2
W1 W2 <E> W3 W4
W1 <E> W2 W3
<E> W1 W2 W3 W4
Entity mention
[Ann Stephens]
[quarter said @ENTITY , president]
[said @ENTITY , president]
[@ENTITY , president of the]
... profits grew by 10% in the last quarter said Ann Stephens , the president of the ...

Figure 1: Architecture of the ladder network (Rasmus et al., 2015) (left), and network initialization of input vector X for the NEC task, where X captures the entity mention modeled and all the patterns matching it in the corresponding sentence (right).

5 Ladder Networks

Ladder networks (LN) (Rasmus et al., 2015) is a neural network architecture designed to use unsupervised learning as a scaffolding for the supervised task. It is a denoising autoencoder (DAE) with noise introduced in every layer. It consists of two sets of encoders, a clean one and another corrupted with noise, and a decoder. In addition, there are skip connections between the encoder and decoder (see left part of Figure 1). The ladder network is defined as follows:

$$\tilde{X}, \tilde{Z}^{(1)}, \ldots \tilde{Z}^{(L)}, \tilde{y} = f_{corr}(X) \tag{4}$$

$$X, Z^{(1)}, \ldots Z^{(L)}, y = f_{clean}(X) \tag{5}$$

$$\hat{X}, \hat{Z}^{(1)}, \ldots \hat{Z}^{(L)} = g(\tilde{Z}^{(1)}, \ldots \tilde{Z}^{(L)}) \tag{6}$$

where X, $\tilde{X}$ and $\hat{X}$ is an input datapoint, its corrupted version, and its reconstruction, respectively; $Z^{(l)}$ and $\tilde{Z}^{(l)}$ are clean and corrupted hidden representations in the l-th layer; and, lastly, y, $\tilde{y}$ are the clean and corrupted activations, converted to a probability distribution over the label set (using a softmax layer).

For our NEC task, X is the concatenation of an entity mention and its context embedding vectors, and y represents the mention's label (e.g., `person`). We initialize the words in the entities and patterns around them with pre-trained word embeddings. To obtain a single embedding for an entity mention and its context we: (a) average word embeddings to obtain a single embedding for the entity mention and each of its patterns; and (b) average the resulting pattern embeddings to produce the embedding of the corresponding context. We then concatenate the mention's embedding and context embedding to be given as input to the ladder network. This process is depicted schematically in the right part of Figure 1. We introduce noise by adding a random Gaussian to the pre-trained embeddings with a fixed mean and standard deviation.

The objective function is a combination of a supervised training cost and unsupervised reconstruction costs at each layer (including the hidden layers):

$$\begin{aligned} Cost = -\sum_{n=1}^{N} logP(\tilde{y_n} = y_n^*|X_n) + \\ \sum_{n=N+1}^{M} \sum_{l=1}^{L} \lambda_l ReconstCost(Z_n^{(l)}, \hat{Z}_n^{(l)}) \end{aligned} \tag{7}$$

where the first term is the supervised cross-entropy based on the N labeled datapoints $(X_1, y_1^*), (X_2, y_2^*), \ldots (X_N, y_n^*)$, and the second term is the reconstruction loss on the M unlabeled datapoints $X_{N+1}, X_{N+2}, \ldots X_{N+M}$, for each layer l. Typically $M \gg N$.

Pezeshki et al. (2016) analyze the different architectural aspects of LN and note that the lateral connections and corresponding reconstruction costs (second term in the above cost function) are critical for

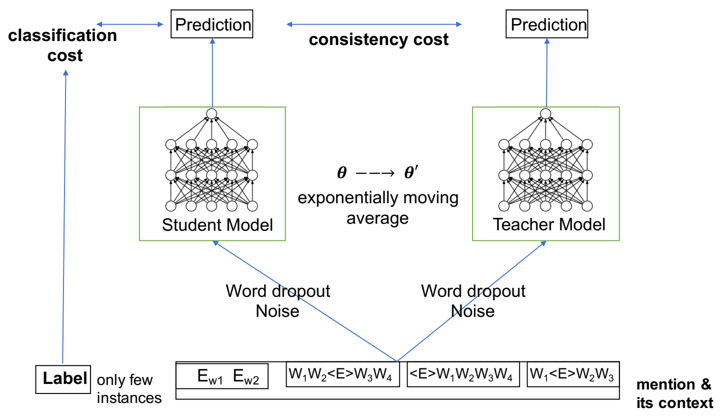


Figure 2: Mean Teacher architecture for the named entity classification task. E_{wi} are words contained in the current entity mention; W_i are words that occur in a pattern that matches the entity mention; and <E> is a placeholder token to indicate where the entity mention occurs inside a pattern.

semi-supervised learning. In other words, it is important for unlabeled data to be used for regularization to be able to learn good abstractions in the different layers. We have similar observations for the NEC task (see Experiments).[1]

6 Mean Teacher Framework

The mean teacher framework (Tarvainen and Valpola, 2017) belongs to a general class of teacher-student algorithms that learn with unlimited unlabeled data and limited supervision. The broad motivation behind MT is provided by the shortcomings of auto-encoders, which suffer from confirmation bias in semi-supervised settings, i.e., when they use their own predictions instead of gold labels (Tarvainen and Valpola, 2017; Laine and Aila, 2016). To mitigate confirmation bias, the teacher weights in MT are averaged over the training epochs, akin to the averaged perceptron. This provides a more robust scaffolding that the student model can rely on during training when gold labels are not available (Tarvainen and Valpola, 2017).

The MT architecture is summarized in Figure 2. It consists of two models with identical architectures where one is termed the teacher, and the other the student. The weights in the teacher network are set through an exponential moving average of the student weights. The input to both of these models is the same datapoint, which is augmented by some noise that is specific to each model. In the case of our NEC task, we introduce noise through random word dropout in the patterns that apply to an entity mention.

The MT algorithm is designed for semi-supervised tasks, where only a few of the data points are labeled. The cost function is a combination of two different type of costs: classification and consistency. The classification cost applies to supervised data points, and can be instantiated with any supervised cost function (e.g., we used categorical cross-entropy). The consistency costs 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:

$$J(\theta) = \mathrm{E}_{x,\eta',\eta}\big[\|f(x,\theta',\eta') - f(x,\theta,\eta)\|^2\big] \tag{8}$$

where this function is defined as the expected distance between the prediction of the teacher model (with weights θ' and noise η') and the prediction of the student model (with weights θ and noise η).

The student weights are updated via back propagation. The teacher weights are deterministically updated after each mini batch of gradient descent on the student, through an exponentially weighted moving average (EMA) from the previous iterations of the student, given by the following equation:

[1]LNs for semi-supervised named entity classification was presented by our earlier work (Nagesh and Surdeanu, 2018). In this work, we compare LNs with MT and provide a more in-depth analysis of these frameworks under the umbrella of teacher-student architectures.

$$\theta_t^{'} = \alpha\theta_{t-1}^{'} + (1-\alpha)\theta_t \tag{9}$$

where $\theta_t^{'}$ is defined at training step t as the EMA of successive weights θ (Tarvainen and Valpola, 2017).

We explored two different architectures for the student model:

Simple model (MT_simple): In this model, we used bag-of-word averaging to obtain embeddings for entity mentions and the patterns matching them. In particular, we: (a) average word embeddings to obtain a single embedding for the entity mention and each of its patterns; and (b) average the resulting pattern embeddings to produce the embedding of the corresponding context. We then concatenate the mention's embedding and context embedding to be given as input to the mean teacher framework. This process is depicted schematically in the right part of Figure 1. This replicates the input processing in LN.

Custom Embedding model (MT_custom): We use a bidirectional LSTM layer on top of the word embedding layer. We train a biLSTM on both the entity tokens and a separate LSTM for each of the pattern tokens. We average each of pattern LSTM representations and concatenate the entity and pattern representations as depicted in the right part of Figure 1. The addition of the LSTM layer, enables this model learn custom embeddings. These embeddings are further analyzed in Section 7.

For both student and teacher models, we initialized the words in the entity mentions and corresponding patterns with pre-trained word embeddings. We have a *linear-ReLU-linear* fully connected layers from the embedding layer before computing the loss.

7 Experiments

7.1 Experimental Setup[2]

Datasets: We used two datasets, the CoNLL-2003 shared task dataset (Tjong Kim Sang and De Meulder, 2003), which contains 4 entity types, and the OntoNotes dataset (Pradhan et al., 2013), which contains 11[3], both of which are benchmark datasets for supervised named entity recognition (NER). These datasets contain marked entity boundaries with labels for each marked entity. Here we only use the entity boundaries but *not* the labels of these entities during the training of our bootstrapping systems. To simulate learning from large texts, we ran the actual experiments on the *train* partitions.[4]

Baselines: We compared against two baselines:

Explicit Pattern-based Bootstrapping (EPB): This is our implementation of the state-of-the-art bootstrapping system of Gupta and Manning (2015), adapted to NEC. The algorithm grows a pool of known entities and patterns for each category of interest, from a few seed examples per category, by iterating between pattern promotion and entity promotion. The former is implemented using a ranking formula driven by the point-wise mutual information (PMI) between each pattern with the corresponding category; the top ranked patterns are promoted to the pattern pool in each iteration. The latter component promotes entities using a classifier that estimates the likelihood of an entity belonging to each class. Our feature set includes, for each category c: (a) edit distance between the candidate entity e and known entities for c; (b) the PMI (with c) of the patterns in the pool of c that matched e in the training documents; and (c) similarity between e and entities in c's pool in some semantic space.[5] Entities classified with the highest confidence for each class are promoted to the corresponding pool after each epoch.

[2]For ease of reproducibility, we release the code for the various semi-supervised learning algorithms at `https://github.com/clulab/releases/tree/master/coling2018-ssl-nec`

[3]We excluded numerical categories such as `DATE`.

[4]Although we run the evaluation on the train dataset, we do not use the labels present in it, as it is a semi-supervised task. In general, in any machine learning dataset, usually the test dataset is quite small compared to train. Since we wanted to analyze the performance in a larger context, we used the train dataset (without the labels).

[5]We used pre-trained word representations, averaged for multi-word entities, to compute cosine similarities between pairs of entities.

	CoNLL	*Ontonotes*
EPB	40.75	21.07
LP	26.07	19.40
Emboot	67.97	45.20
Ladder	68.92 (±2.9)	65.20 (±3.2)
MT_simple	69.50 (±2.7)	51.10 (±4.8)
MT_custom	66.81 (±3.5)	36.17 (±2.5)

Table 1: Overall Accuracies (entity-based)

	CoNLL	*Ontonotes*
Ladder	71.58 (± 1.6)	72.04 (± 0.7)
MT_simple	73.91 (± 3.4)	66.27 (± 1.9)
MT_custom	68.14 (± 4.3)	64.04 (± 3.8)

Table 2: Overall Accuracies (mention–based)

Label Propagation (LP): We used the implementation available in the `scikit-learn` package of the LP algorithm (Zhu and Ghahramani, 2002).[6] In each bootstrapping epoch, we run LP, select the entities with the lowest entropy, and add them to their top category. Each entity is represented by a feature vector that contains the co-occurrence counts of the entity and each of the patterns that matches it in text.[7]

Settings: For each entity mention, we consider a n-gram window of size 4 on either side as a pattern. We initialized the mention and contexts embeddings input to either the ladder network or the mean teacher as well as the baseline systems with pre-trained embeddings from Levy and Goldberg (2014b) (size 300d) as this gave us improved results on the baseline compared to vanilla `word2vec` initialization. We used a 600d dimensional embedding for each datapoint (300 each from entity and context concatenated). We used a 3-layer ladder network with dimensions 600-500-K where K is the number of labels present in the dataset. Further, we used a noise of 0.3 for the corrupted encoder and reconstruction cost for the 3-layers were 1000-10-0.1. In the custom embeddings architecture of the mean teacher framework we used an bi-directional LSTM of size 200d on the entities and the each of the patterns in the context. The hidden size of fully connected layer was set to 300. We used a random word dropout of size 1 on the patterns to be input to the student and the teacher. Following Tarvainen and Valpola (2017), in the MT runs, we set the consistency cost weight to 1 and consistency-rampup to 5. We set the supervised examples (mentions along their corresponding contexts and labels) randomly. For CoNLL we used 40 and Ontonotes 440 examples, with equal representation from their labels' set.

Note that both MT and LN classify individual mentions, whereas Emboot and the two baselines classify entities. To facilitate the comparison between MT and LN and the baselines, for MT and LN we generate entity-level predictions by averaging the scores of the top predictions of all mentions corresponding to the same entity, and then select the label with the highest score for each entity. Further, to create the precision/throughput curves we we sorted the predictions returned by the LN and MT in decreasing order of their activation scores, and introduce multiple cutout thresholds to generate the precision/throughput points for the curves. We ran the baselines until they predicted labels for all the entities. For the baselines, in each iteration we promoted 100 entities per category.[8]

To compare with the baselines, which classify entities rather than mentions, we sorted the predictions returned by the LN and MT in decreasing order of their activation scores and chose the most confident entity label (when all its mention scores were averaged). For each of the LN and MT results, we report averaged results of 5 random runs (along with the standard deviation).

7.2 Results and Observations

Tables 1 and 2 shows the overall accuracies of the various systems on entity-based and mention-based evaluations, respectively. The key observation is that there is a consistent improvement in performance in approaches that are based on representation learning (namely, Emboot [9], LN and MT) on both datasets compared to traditional bootstrapping techniques (EPB and LP). Furthermore, we observe that one-shot learning systems such as Ladder and MT have a clear edge over the embedding based iterative

[6] `http://scikit-learn.org/stable/modules/generated/sklearn.semi_supervised.LabelPropagation.html`

[7] We experimented with other feature values, e.g., pattern PMI scores, but all performed worse than raw counts.

[8] We also ran a cautious approach of promoting 10 entities per category per iteration and noticed that the former had slightly better performance.

[9] Note: The Emboot results are for 20 epochs only. As running more number of epochs to cover the entire dataset does not complete even after a long time and sometimes results in memory overflow in the current implementation.

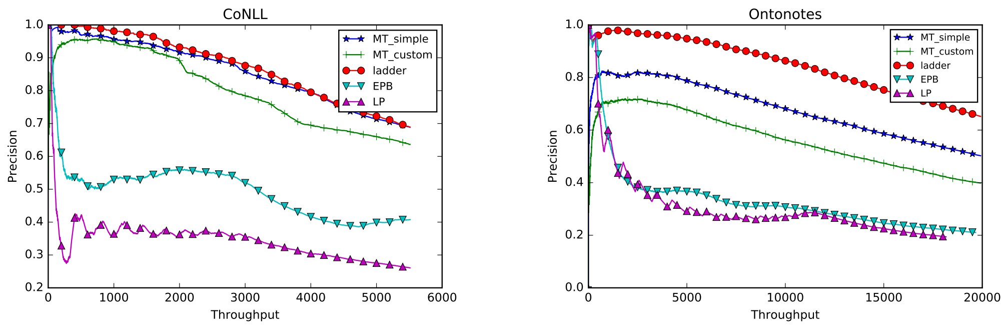


Figure 3: Overall results on the CoNLL (left) and Ontonotes (right) datasets. Throughput is the number of entities classified, and precision is the proportion of entities that were classified correctly.

bootstrapping (Emboot). To our knowledge, the results in Table 2 are the new state of the art for this semi-supervised NEC task.

While MT and LN outperform Emboot on both datasets, it is yet not fully clear why. As mentioned, MT and LN use one-shot learning whereas Emboot is iterative and, thus, more prone to semantic drift. However, a second potential cause for this difference is that MT and LN assemble their input representation from word-level embeddings, whereas Emboot learns from scratch embeddings for atomic entities and patterns, which forces it to work with sparser information. We leave this analysis, as well as efforts to align these three frameworks more closely, as future work.

Figure 3 shows the precision vs. throughput curves for the baselines, LN and MT approaches. We see that on both the datasets the one-shot learning methods outperform the iterative baselines by a large margin. Further we notice that both LN and MT variants are reasonably stable for most of the curve whereas EPB degrades quickly. Bootstrapping systems inherently suffer from semantic drift: as the iterations progress the system begins to drift into a different semantic space due to incomplete statistics and ambiguity (Komachi et al., 2008).

MT outperforms slightly LN on the CoNLL dataset, whereas LN performs better than MT on OntoNotes. We speculate that the latter result is caused by the fact that the space of hyper parameters were not sufficiently explored in the MT framework. For example, we ran LN for approximately 200 iterations, and we ran the MT experiments for 20 epochs. We will investigate a larger hyper parameter space in future work. However, the comparable performance of the two approaches is an encouraging result for MT, which is the simpler architecture of the two. With MT, we can "plug in" any task simply by generating a different input vector X (and, potentially, a different scheme to insert noise), whereas LN requires an adaptation of the entire architecture due to the tight connections between teacher and student.

Qualitative analysis: In Figure 4, we plot the t-SNE (van der Maaten and Hinton, 2008) projections of the learned custom embeddings from the MT framework. We contrast these projections with the t-SNE projections of the entities constructed by averaging the vectors present in the entity tokens, as provided by the pre-trained embeddings we used to initialize the various models. It is interesting to note that we can see clear clustering of the entities in both datasets for MT. The number of clusters is roughly equal to the number of classes present in the datasets. This is especially accentuated on the OntoNotes dataset.Upon closer observation on a small number of data points, we could ascertain that the clusters were semantically meaningful. On the other hand, we do not see such a clustering of the pre-trained embeddings (word2vec). This opens the gates for further investigation of these custom embeddings to construct interpretable deep learning models.

Amount of training data vs. accuracy: Table 3 lists the accuracies of LN and MT_simple approaches on all the data points, as we varied the amount of supervision. As expected, as we increase the amount of supervision, we observe improvements in accuracy. More importantly, the table shows both LN and MT outperform the overall accuracy of EPB (right-most points in Figure 3) with much fewer annotations (e.g., with 55 annotations in OntoNotes, LN and MT outperform the performance of EPB with 550

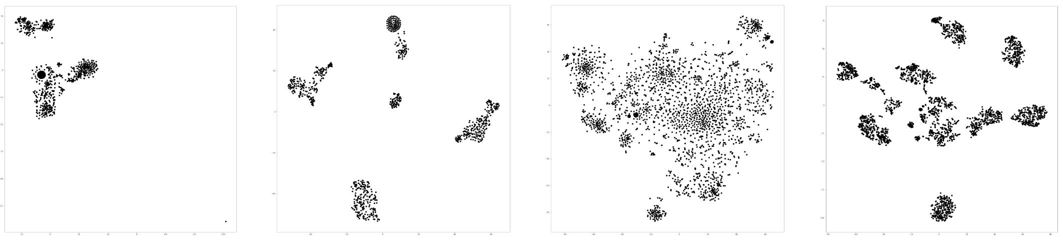


Figure 4: t-SNE projections of the learned embeddings of four models: (i) pre-trained word2vec on the CoNLL dataset; (ii) the custom MT model on CoNLL; (iii) pre-trained word2vec on OntoNotes; (iv) custom MT on OntoNotes.

CoNLL				*OntoNotes*			
Num. labels	% Train	*Ladder*	*MT*	*Num. labels*	% Train	*Ladder*	*MT*
20	0.152	46.46	72.74	55	0.082	26.04	57.05
40	0.303	66.46	80.53	110	0.164	48.53	58.40
80	0.606	75.37	80.96	220	0.328	59.66	58.87
160	1.212	81.11	82.06	440	0.656	73.10	61.72
320	2.424	80.94	82.58	880	1.313	73.58	61.96
640	4.848	82.51	82.60	1760	2.626	73.23	67.38
1280	9.696	81.22	83.31	3520	5.253	73.77	70.52
2560	19.393	81.34	83.47	7040	10.507	73.31	72.65
5120	38.787	81.26	84.88	14080	21.014	82.47	73.50
10240	77.575	81.91	86.30	28160	42.029	83.32	75.91

Table 3: Number of labeled examples used for training and corresponding accuracy on the two datasets.

annotated examples).

8 Conclusion

To our knowledge, this is the first work that rigorously analyzes several recent semi-supervised representation-learning approaches in the context of an information extraction task – named entity classification, in particular. To achieve this, we adapted two teacher-student methods that were initially proposed for image processing to the named entity classification task, and compared them against an iterative representation learning that learns custom embeddings for the underlying data, as well as other semi-supervised learning approaches that do not rely on representation learning. Our analysis, which was performed on two NEC datasets, highlighted several important observations, which, we believe, should inform the design of future semi-supervised IE systems. First, all representation learning methods investigated outperform state-of-the-art semi-supervised methods that do not rely on representation learning. Second, one-shot learning methods outperform iterative approaches, which suggests that one-shot learning is a practical solution to control for semantic drift. Third, the mean teacher framework performs comparatively similar to ladder networks, despite its simplicity and independence of the underlying task-specific model. These results advocate for a modular take on semi-supervised IE, where the learning is performed by a generic MT framework, which is then extended for specific tasks through independent components that model the corresponding data and task.

As part of future work, we would like to explore other tasks such as fine-grained entity typing, where the number of labels are an order of magnitude larger. Further, another interesting avenue for further research is to perform semi-supervised relation extraction, which is a significantly more challenging task and would be greatly benefited by such techniques.

Acknowledgements

Dr. 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. This work was funded by the DARPA Big Mechanism program under ARO contract W911NF-14-1-0395 and by the Bill and Melinda Gates Foundation HBGDki Initiative.

References

Steven Abney. 2007. *Semisupervised Learning for Computational Linguistics*. Chapman & Hall/CRC, 1st edition.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, COLT' 98, pages 92–100, New York, NY, USA. ACM.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010a. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI 2010)*.

Andrew Carlson, Justin Betteridge, Richard C Wang, Estevam R Hruschka Jr, and Tom M Mitchell. 2010b. Coupled semi-supervised learning for information extraction. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 101–110. ACM.

Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Olivier Delalleau, Yoshua Bengio, and Nicolas Le Roux. 2005. Efficient non-parametric function induction in semi-supervised learning. In Robert G. Cowell and Zoubin Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (AISTATS'05)*, pages 96–103. Society for Artificial Intelligence and Statistics, January.

Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of NAACL*.

Sonal Gupta and Christopher D Manning. 2014. Improved pattern learning for bootstrapped entity extraction. In *CoNLL*, pages 98–108.

Sonal Gupta and Christopher D. Manning. 2015. Distributed representations of words to guide bootstrapped entity classifiers. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*.

Mamoru Komachi, Taku Kudo, Masashi Shimbo, and Yuji Matsumoto. 2008. Graph-based analysis of semantic drift in espresso-like bootstrapping algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 1011–1020, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

Omer Levy and Yoav Goldberg. 2014a. Dependency-based word embeddings. In *ACL (2)*, pages 302–308.

Omer Levy and Yoav Goldberg. 2014b. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, Baltimore, Maryland, June. Association for Computational Linguistics.

Tara McIntosh and James R Curran. 2008. Weighted mutual exclusion bootstrapping for domain independent lexicon and template acquisition. In *Proceedings of the Australasian Language Technology Association Workshop*, volume 2008.

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. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Ajay Nagesh and Mihai Surdeanu. 2018. Keep your bearings: Lightly-supervised information extraction with ladder networks that avoids semantic drift. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 352–358. Association for Computational Linguistics.

Arvind Neelakantan and Michael Collins. 2015. Learning dictionaries for named entity recognition using minimal supervision. *CoRR*, abs/1504.06650.

Mohammad Pezeshki, Linxi Fan, Philemon Brakel, Aaron Courville, and Yoshua Bengio. 2016. Deconstructing the ladder network architecture. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2368–2376, New York, New York, USA, 20–22 Jun. PMLR.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Bjrkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using ontonotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152, Sofia, Bulgaria, August. Association for Computational Linguistics.

Delip Rao and Deepak Ravichandran. 2009. Semi-supervised polarity lexicon induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 675–682, Stroudsburg, PA, USA. Association for Computational Linguistics.

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, *Advances in Neural Information Processing Systems 28*, pages 3546–3554. Curran Associates, Inc.

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

Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In *Proceedings of the national conference on artificial intelligence*, pages 1044–1049.

Rebecca Sharp, Mihai Surdeanu, Peter Jansen, Peter Clark, and Michael Hammond. 2016. Creating causal embeddings for question answering with minimal supervision. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 1195–1204. Curran Associates, Inc.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 142–147. Edmonton, Canada.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *EMNLP*, volume 15, pages 1499–1509. Citeseer.

Kristina Toutanova, Xi Victoria Lin, Wen-tau Yih, Hoifung Poon, and Chris Quirk. 2016. Compositional learning of embeddings for relation paths in knowledge bases and text. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 1434–1444.

Marco A. Valenzuela-Escárcega, Ajay Nagesh, and Mihai Surdeanu. 2018. Lightly-supervised representation learning with global interpretability. *CoRR*, abs/1805.11545.

Harri Valpola. 2014. From neural PCA to deep unsupervised learning. *CoRR*, abs/1411.7783.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 1096–1103, New York, NY, USA. ACM.

Roman Yangarber. 2003. Counter-training in discovery of semantic patterns. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Shuangfei Zhai and Zhongfei Zhang. 2015. Semisupervised autoencoder for sentiment analysis. *CoRR*, abs/1512.04466.

Yizhe Zhang, Dinghan Shen, Guoyin Wang, Zhe Gan, Ricardo Henao, and Lawrence Carin. 2017. Deconvolutional paragraph representation learning. *CoRR*, abs/1708.04729.

X. Zhu and Z. Ghahramani. 2002. Learning from labeled and unlabeled data with label propagation. Technical Report CMU-CALD-02-107, Carnegie Mellon University.