The Unreasonable Effectiveness of Word Representations for Twitter Named Entity Recognition

Colin Cherry and **Hongyu Guo**
National Research Council Canada
first.last@nrc-cnrc.gc.ca

Abstract

Named entity recognition (NER) systems trained on newswire perform very badly when tested on Twitter. Signals that were reliable in copy-edited text disappear almost entirely in Twitter's informal chatter, requiring the construction of specialized models. Using well-understood techniques, we set out to improve Twitter NER performance when given a small set of annotated training tweets. To leverage unlabeled tweets, we build Brown clusters and word vectors, enabling generalizations across distributionally similar words. To leverage annotated newswire data, we employ an importance weighting scheme. Taken all together, we establish a new state-of-the-art on two common test sets. Though it is well-known that word representations are useful for NER, supporting experiments have thus far focused on newswire data. We emphasize the effectiveness of representations on Twitter NER, and demonstrate that their inclusion can improve performance by up to 20 F1.

1 Introduction

Named entity recognition (NER) is the task of finding rigid designators as they appear in free text and classifying them into coarse categories such as *person* or *location* (Nadeau and Sekine, 2007). NER enables many other information extraction tasks such as relation extraction (Bunescu and Mooney, 2005) and entity linking (Ratinov et al., 2011).

There is considerable excitement at the prospect of porting information extraction technology to social media platforms such as Twitter. Social media reacts to world events faster than traditional news sources, and its sub-communities pay close attention to topics that other sources might ignore. An early example of the potential inherent in social information extraction is the Twitter Calendar (Ritter et al., 2012), which detects upcoming events (concerts, elections, video game releases, etc.) based on the anticipatory chatter of Twitter users. Unfortunately, processing social media text presents a unique set of challenges, especially for technologies designed for newswire: Twitter posts are short, the language is informal, capitalization is inconsistent at best, and spelling variations and abbreviations run rampant.

Armed with an affordable training set of 1,000 annotated tweets, we establish a strong baseline for Twitter NER using well-understood techniques. We build two unsupervised word representations in order to leverage a large collection of unannotated tweets, while a data-weighting technique allows us to benefit from annotated newswire data. Taken together, these two simple ideas establish a new state-of-the-art for both our test sets. We rigorously test the impact of both continuous and cluster-based word representations on Twitter NER, emphasizing the dramatic improvement that they bring. We also bring the experimental methodology of the domain adaptation community to Twitter NER, testing in-domain, out-of-domain and combined training scenarios, and revealing that it is not trivial to benefit from out-of-domain training data. Finally, an error analysis helps us begin to understand which social media challenges are being addressed by our adaptations, and which problems persist.

2 Background

Our work builds on a long line of research in discriminative tagging (Collins, 2002), and its application to named entity recognition (McCallum and Li, 2003). Our baseline tagger draws inspiration from Sarawagi and Cohen (2004), who introduce the no-

Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL, pages 735–745,
Denver, Colorado, May 31 – June 5, 2015.

tion of semi-Markov tagging for NER, and from de Bruijn et al. (2011), who apply a similar tagger to clinical information extraction.

A number of previous studies have closely examined the use of word representations in NER, where one leverages unlabeled data to build features that help the tagger generalize across similar words. Miller et al. (2004) introduce this idea and provide the framework to build representation features from word clusters, while Lin and Wu (2009) extend this technique with phrases and sheer masses of unlabeled data. Turian et al. (2010) introduce continuous vectors as alternative word representations, and provide several experiments comparing these with clusters. Recently, Passos et al. (2014) have shown how continuous representations can be tailored to NER with a combination of context- and gazetteer-aware objectives. All of these studies employ representations only in newswire scenarios. Ratinov and Roth (2009) investigate cluster representations in a Web NER task, but the performance of their baseline indicates that it is not nearly so drastic a domain shift as our Twitter task.

2.1 Adapting to Social Media

There has been much recent activity in adapting NLP tools for social media. Ritter et al. (2011) collect training data and adapt tools for a number of tasks, including part-of-speech (POS) tagging, shallow parsing and NER. Owoputi et al. (2013) extends a line of research on building robust POS taggers for Twitter, and share our focus on the utility of word representations in this domain.

Liu et al. (2011) carry out the first study to specifically examine NER on Twitter. They use a nearest-neighbour word classifier stacked with a CRF, along with a boot-strapping scheme for semi-supervised learning. Interestingly, they find no utility in using cluster-based word representations, perhaps because their model directly accounts for a type's global context with bag-of-word features. Ritter et al. (2011) also examine Twitter NER, developing a semi-supervised technique that uses labeled LDA to project information from Freebase gazetteers onto unlabeled tweets. Plank et al. (2014) suggest a distant-supervision scheme, creating artificial training data by projecting reliable NER tags from web pages onto the tweets that link to those pages.

Fromreide et al. (2014) and Plank et al. (2014) point out that NER performance can be overestimated when a system is tested on data extracted from the same pool as its training data. Temporal effects and annotation biases can result in gains that disappear when shifting to another test set. We follow their lead by testing on data that was annotated independently from our training data.

3 Methods

Our named entity recognizer is a discriminative, semi-Markov tagger, trained online using large-margin updates. It differs from word-based CRF systems in three ways: its inference algorithm, its tag structure, and its learning algorithm. This tagger allows us to develop new systems quickly, but it is important to emphasize that the adaptation strategies described later in this section can just as easily be applied to word-based CRFs.

Semi-Markov Inference

Sarawagi and Cohen (2004) describe a straightforward extension to the Viterbi algorithm that enables the tagging of contiguous phrases instead of words. Because each phrasal entity is tagged as a unit, we can recover entity boundaries without distinguishing between *Begin* and *Inside* tags, leaving the tagger to track only entity classes and *Outside* tags. This in turn allows us to run our tagger without Markov features. Since most entities are surrounded by *Outside* tags, conditioning on previous tag assignments has only limited utility. Finally, our phrasal tags enable useful features that consider entire entities, such as phrase-identity indicators.

Phrasal and Word-level Tags

In word-based models, it is beneficial to not only identify words that *Begin* entities, but also those that are in the middle (*Inside*) or at the end of entities (*Last*), as well as entities that consist of exactly one *Unique* word (Ratinov and Roth, 2009). Since we tag entire phrases at once, we can easily assign each word in the phrase to one of these four entity-relative positions. Therefore, even though our tagger tracks only entity class, its word-level features are annotated as if we maintained a full *BILUO* tag set.

Passive-Aggressive Learning

We train our model with a structured version of the Passive-Aggressive (PA) algorithm (Crammer et al., 2006). The benefits of using PA in place of a CRF are that we require only Viterbi inference, and memory requirements are minimized, as we update the model one training sentence at a time.

PA is an online, large-margin learning algorithm that attempts to separate correct sequences from incorrect ones by a margin of 1. For each update to the weight vector w, we select a training sentence x and its gold-standard tag sequence y. We use dynamic programming to search for a response $\hat{y}$ that maximizes the structured hinge loss:[1]

$$\hat{y} = \arg\max_{y' \neq y} = \left[1 + w^{\mathrm{T}}\big(\Phi(x, y') - \Phi(x, y)\big)\right] \tag{1}$$

where $\Phi()$ maps an (x, y) pair to a feature vector. If the loss is greater than 0, we update our model:

$$w = w + \tau\big(\Phi(x, y) - \Phi(x, \hat{y})\big) \tag{2}$$

where τ is an adaptive learning rate that scales the update to the smallest step size that achieves 0 loss:

$$\tau = \min\left(C, \frac{1 + w^{\mathrm{T}}\big(\Phi(x, \hat{y}) - \Phi(x, y)\big)}{||\Phi(x, y) - \Phi(x, \hat{y})||}\right) \tag{3}$$

C is a hyper-parameter that truncates large steps to prevent over-fitting. It is related to the C-parameter of an SVM (Martins et al., 2010). To further guard against over-fitting, we use the average of all vectors w seen during training when tagging new text (Collins, 2002).

Features

The feature function $\Phi(x, y)$ must decompose into the semi-Markov dynamic program:

$$\Phi(x, y) = \sum_{(s,t,y_j) \in D(x,y)} \phi(s, t, y_j, x) \tag{4}$$

where D is a derivation decomposing (x, y) into J entity-tag assignments (s, t, y_j), each asserting that the phrase $x_s \ldots x_{t-1}$ is assigned the tag y_j. Tagged spans are non-overlapping, and to eliminate spurious ambiguity, constrained so that *Outside* can tag only single-word spans ($t = s + 1$).

Phrase:
$[y_j]$, $[y_j, x_s \ldots x_{t-1}]$, $[y_j, \mathit{lc}(x_s \ldots x_{t-1})]$, $[y_j, \mathit{ss}(x_s \ldots x_{t-1})]$
Word, for each i s.t. $s \leq i < t$:
$\{[y_j, x_{i+k}, k]\}_{k=-2}^{2}$, $\{[y_j, er_{s,t}(i), x_{i+k}, k]\}_{k=-2}^{2}$, $\{[y_j, \mathit{lc}(x_{i+k}), k]\}_{k=-2}^{2}$, $\{[y_j, er_{s,t}(i), \mathit{lc}(x_{i+k}), k]\}_{k=-2}^{2}$, $\{[y_j, \mathit{ss}(x_{i+k}), k]\}_{k=-2}^{2}$, $\{[y_j, er_{s,t}(i), \mathit{ss}(x_{i+k}), k]\}_{k=-2}^{2}$, $\{[y_j, \mathit{pf}(n, x_i)]\}_{n=1}^{3}$, $\{[y_j, er_{s,t}(i), \mathit{pf}(n, x_i)]\}_{n=1}^{3}$, $\{[y_j, \mathit{sf}(n, x_i)]\}_{n=1}^{3}$, $\{[y_j, er_{s,t}(i), \mathit{sf}(n, x_i)]\}_{n=1}^{3}$,

Table 1: Baseline features $\phi(s, t, y_j, x)$. [*str*] stands for an indicator feature with the name *str*; *lc*() maps a string onto its lowercased form; *ss*() maps a string onto its word shape ("Apple Inc." becomes "Aa Aa."); *pf*(n, x_i) and *sf*(n, x_i) are n-character prefixes and suffixes of x_i; and $er_{s,t}(i)$ maps an absolute sentence position i ($s \leq i < t$) to a relative entity position drawn from $\{B, I, L, U\}$.

Our baseline feature set, shown in Table 1, closely mimics the set proposed by Ratnaparkhi (1996), covering word identity, prefixes, suffixes and surrounding words. It has been augmented with phrase-identity indicators and hierarchical word-level tags. These conjoin the entity class y_j with the word's entity-relative position, backing off to y_j alone. Most features look only at a single word x_i, which improves efficiency by allowing the tagger to re-use word-level scores across many phrasal tags.

There are some standard NER features that we chose not to include. We follow Lin and Wu (2009) in omitting POS tags and gazetteers in order to reduce our dependence on linguistic resources. We expect similar information to be provided by unsupervised word representations, and we test this assumption in Section 5.2. We omit context aggregation, which accounts for the repetition of entities (Ratinov and Roth, 2009), because Twitter's short message length reduces the utility of document-level features.

3.1 Word Representations

Our primary tool for domain adaptation will be unsupervised word representations, which convey information about a word's distributional profile.

[1] This can be done by running a 2-best tagger. If the 1-best answer is not correct ($y' \neq y$), then it maximizes the loss, otherwise, the 2-best answer maximizes the loss.

Brown clusters, for each i s.t. $s \leq i < t$:
$\{[y_j, \mathit{brn}(n, x_i), n]\}_{n \in \{2,4,8,12\}}$,
$\{[y_j, \mathit{er}_{s,t}(i), \mathit{brn}(n, x_i), n]\}_{n \in \{2,4,8,12\}}$

Word vectors, for each i s.t. $s \leq i < t$:
$\{[y_j, n] = \mathit{w2v}(n, x_i)\}_{n=1}^{300}$,
$\{[y_j, \mathit{er}_{s,t}(i), n] = \mathit{w2v}(n, x_i)\}_{n=1}^{300}$

Table 2: Word representation features in $\phi(s, t, y_j, x)$. $\mathit{brn}(n, x_i)$ maps a word x_i to the first n bits of its Brown cluster bit sequence. $\mathit{w2v}(n, x_i)$ maps x_i to the n^{th} component of its word vector, and $[\mathit{str}] = v$ stands for a real-valued feature with name *str* and value v.

Brown Clusters

The Brown clustering algorithm assigns types to a deterministic, hierarchical clustering, which has been trained to optimize the likelihood of a first-order, class-based language model (Brown et al., 1992). The clusters capture both syntactic and semantic regularities, and have been shown to perform well as unsupervised part-of-speech taggers (Blunsom and Cohn, 2011).

The clusters are organized into a binary tree structure; therefore, each cluster can be represented as a bit string that encodes the branching decisions required to reach its leaf from the root. By truncating the bit string at different prefix lengths, one can access different granularities of clusters. Cluster membership can then be used to create indicators similar to the baseline's word identity features. This results in two feature templates, shown in Table 2.[2]

This technique has been previously applied to both newswire NER (Miller et al., 2004; Turian et al., 2010; Passos et al., 2014) and Twitter NER (Ritter et al., 2011; Plank et al., 2014). But previous work on Twitter NER has not directly tested the impact of Brown clusters; instead, they generally appear as part of an adapted baseline.

Word Vectors

An alternative word representation maps each word type deterministically to a low-dimensional continuous vector space. This technique was originally used as the bottom layer for continuous-space language models (Bengio et al., 2003), where the type-to-vector mapping can be learned with backpropagation. However, Mikolov et al. (2013) have shown that useful vector representations can be learned more efficiently by eschewing the language-modeling objective. Their skip-gram model, which we adopt here, optimizes for each token, the likelihood of the tokens in a window surrounding it. This training process creates a linear classifier that predicts words conditioned on the central token's vector representation. The classifier and the word vectors are learned simultaneously, but once training is complete, the classifier is usually discarded, leaving only the vectors.

These continuous representations project words into a low-dimensional space. Words that tend to have similar contexts, and therefore similar syntactic and semantic properties, will tend to be near one another in this space. We incorporate these representations into our NER system as real-valued features of each word x_i, as shown in Table 2.

3.2 Data Weighting

Our next tool for domain adaptation is a small pool of in-domain, annotated data. The easiest way to make use of this data is to append it to our large pool of out-of-domain training data, which is what has been done in previous work on Twitter NER (Ritter et al., 2011; Plank et al., 2014). However, we have the strong intuition that greater weight should be placed on the in-domain data.

Assume that for each training pair (x, y) we also have an importance weight η. In our case, all out-of-domain pairs will share one value for η, and all in-domain pairs will share another, higher η. We modify our PA learner to calculate τ using a version of Equation 3 that replaces C with ηC. Unlike scaling τ directly, scaling C has the desirable property of having even high-η examples stop updating at precisely 0 loss, just as if we had duplicated that training example η times (Karampatziakis and Langford, 2010). If we view C as a regularization term, then this modification can also be interpreted as implementing example-specific regularization. Importance weights can also be incorporated into CRFs by modifying their training objective; however, this is not a standard feature of most CRF packages.

[2] We also experimented with templates over clusters and vectors for surrounding words, to no benefit.

Data	Lines	Types	Tokens	# PER	# LOC	# ORG
Fin10 (Train)	1,000	4,865	17,276	192	143	172
Fin10Dev (Test)	1,975	7,734	33,770	325	279	287
Rit11 (Test)	2,394	8,686	46,469	454	377	280
Fro14 (Test)	1,545	5,392	20,666	390	163	200
CoNLL (Train)	14,041	20,752	203,621	6,601	7,142	6,322
Unlabeled Tweets	98M	57M	1,995M	–	–	–

Table 3: Details of our NER-annotated corpora. A *line* is a tweet in Twitter and a sentence in newswire.

4 Experimental Design

Vital statistics for all of our data sets are shown in Table 3. For in-domain NER data, we use three collections of annotated tweets: Fin10 was originally crowd-sourced by Finin et al. (2010), and was manually corrected by Fromreide et al. (2014), while Rit11 (Ritter et al., 2011) and Fro14 (Fromreide et al., 2014) were built by expert annotators. We divide Fin10 temporally into a training set and a development set, and we consider Rit11 and Fro14 to be our test sets. This reflects a plausible training scenario, with train and dev drawn from the same pool, but with distinct tests drawn from later in time. These three data sets were collected and unified by Plank et al. (2014), who normalized the tags into three entity classes: *person* (PER), *location* (LOC) and *organization* (ORG). The source text has also been normalized; notably, all numbers are normalized to NUMBER, and all URLs and Twitter @user names have been normalized to URL and @USER respectively. In the gold-standard, we choose to reverse a tagging normalization performed by Plank et al. (2014), who had post-processed the data so that all @user names are tagged as PER. These tags are trivial to replicate, and we found that they inflate scores quite dramatically. Therefore, all @user names are untagged in both the gold standard and our system outputs.

We use the CoNLL 2003 newswire training set as a source of out-of-domain NER annotations (Tjong Kim Sang and De Meulder, 2003). The source text has been normalized to match the Twitter NER data, and we have removed the MISC tag from the gold-standard, leaving PER, LOC and ORG.

Finally, we also use a large corpus of unannotated tweets, collected from between May 2011 and April 2012. It has been tokenized by the CMU Twokenizer,[3] but is otherwise unnormalized.

4.1 Hyper-parameter Configuration

Our NER system is trained for 10 epochs with its regularization parameter C set to 0.01.

We train our word vectors with an in-house implementation of word2vec (Mikolov et al., 2013), with vector size set to 300, a hierarchical soft-max objective, down-sampling frequent words at a rate of 0.001, a window-size of 10 tokens, and a minimum frequency count of 10. When run on our unannotated tweets, this produces vector representations for 2.5M types. We generate a random vector, with each component sampled from the standard normal, to use as the representation for any word that did not occur in our unlabeled data, including begin- and end-of-sentence markers. We do not scale the vectors before using them as NER features.

We train Brown clusters on the same data using the implementation by Liang (2005), with 1,000 clusters and a minimum frequency of 10, resulting in cluster assignments for the same 2.5M types.

5 Results

We evaluate our various NER taggers using the CoNLL 2003 metrics: phrase-level precision, recall, and balanced F-measure (F1).

We begin by testing our system on the CoNLL newswire task, both to confirm that our implementation is reasonable, and to help situate the Twitter results that appear later. We train on the unmodified CoNLL training corpus, and report F1 on the CoNLL development and test sets. We compare our baseline to the baseline from Ratinov and Roth (2009) (RR09), and we compare our representation-enhanced system (*+Reps*) to their “All External

[3] http://www.ark.cs.cmu.edu/TweetNLP/

System	Dev F1	Test F1
RR09 Baseline	89.2	83.6
Our Baseline	90.4	84.3
RR09 Base + All External	92.5	88.6
Our Base + Reps	91.6	88.0

Table 4: Performance on newswire (CoNLL) data.

System	Fin10Dev	Rit11	Fro14	Avg
CoNLL	27.3	27.1	29.5	28.0
+ Brown	38.4	39.4	42.5	40.1
+ Vector	40.8	40.4	42.9	41.4
+ Reps	42.4	42.2	46.2	43.6
Fin10	36.7	29.0	30.4	32.0
+ Brown	59.9	53.9	56.3	56.7
+ Vector	61.5	56.4	58.4	58.8
+ Reps	64.0	58.5	60.2	60.9
CoNLL+Fin10	44.7	39.9	44.2	42.9
+ Brown	54.9	52.9	58.5	55.4
+ Vector	58.9	55.2	59.9	58.0
+ Reps	58.9	56.4	61.8	59.0
+ Weights	64.4	59.6	63.3	62.4

Table 5: Impact of our components on Twitter NER performance, as measured by F1, under 3 data scenarios.

Knowledge" system. Both use Brown clusters, but RR09 uses Wikipedia gazetteers where we use word vectors. Results are shown in Table 4.

We achieve broadly comparable scores in both settings. Our external knowledge features are not as useful as theirs, which may be due to our lack of Wikipedia gazetteers, or due to a domain mismatch in our unannotated training data. Their clusters are trained on the 1996 Reuters corpus, a superset of the CoNLL data, matching it in both era and domain. Conversely, our clusters and vectors are both trained on tweets from 2011, so it is somewhat surprising that they help to the extent that they do.

5.1 Performance on Twitter

Our primary results are shown in Table 5, where we compare our word representation and data weighting techniques under three scenarios: training on out-of-domain data only (*CoNLL*), on in-domain data only (*Fin10*), and on both. Our data weighting technique (+*Weights*, see Section 3.2) only applies when we use both training sets. We used Fin10Dev to determine our importance weights, selecting $\eta = 0.01$ for CoNLL and $\eta = 1$ for Fin10. For these experiments, we test Brown clusters (+*Brown*), word vectors (+*Vector*), and both together (+*Reps*).

System	Prec	Rec	F1
CoNLL	43.0	49.8	46.2
Fin10	75.3	50.2	60.2
CoNLL+Fin10	66.0	58.0	61.8
+Weights	73.8	55.4	63.3

Table 6: Precision, recall and F1 on the *Fro14* test set with the *Base+All Reps* feature set.

Our Twitter NER results are much lower than the newswire results from Table 4, with our best Twitter system scoring more than 25 F1 below our best CoNLL system. But the picture would look much worse without word representations, which boost performance in every training scenario. Our best representation-free system lags nearly 20 F1 behind our best system that uses representations.

Looking across scenarios, we note that *CoNLL+Reps* outperforms *CoNLL+Fin10* on 2 out of 3 tests. This is interesting, as it shows that, given the hypothetical choice between collecting 100 million unannotated tweets for word representations, and collecting one thousand annotated tweets for NER training, we are better served by the unannotated data. Of course, it is even better to use both; their combined benefit in *CoNLL+Fin10+Reps* is more than additive.

Across all data scenarios and test sets, Brown clusters help less than word vectors. This contradicts the observations from Turian et al. (2010), who generally found Brown clusters to perform best. This may be because of our domain adaptation scenario, or it could be due to our use of word2vec, which did not exist at the time of the Turian study. The combination of Brown clusters and word vectors is consistently better than using either alone. The two representations track different sorts of information: our use of a large window leads word2vec to build topic-focused vectors (Turney, 2012), while Brown clustering is naturally more local, creating very syntactic, part-of-speech-like clusters. It is easy to see how both types of information can be useful to NER.

Ritter et al. (2011) report that including out-of-

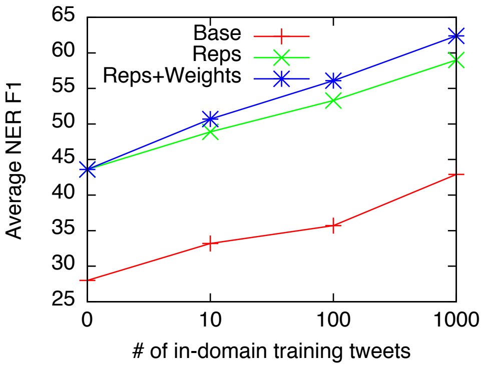

Figure 1: F1 averaged over all 3 test sets as we add Fin10 training data to CoNLL.

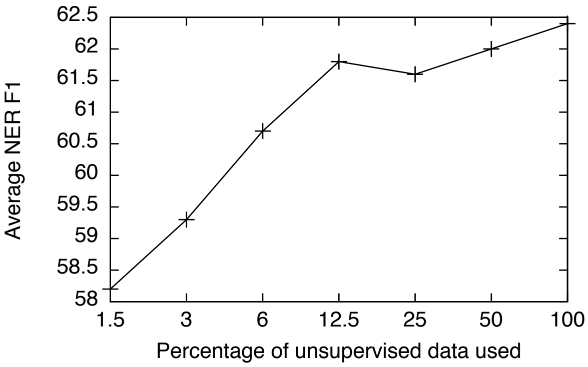

Figure 2: F1 averaged over all 3 test sets as we increase the percentage of tweets used to build representations.

domain data hurts NER performance. Focusing on the lines *Fin10+Reps* and *CoNLL+Fin10+Reps*, we see the same problem. In the presence of word representations, unweighted CoNLL data hurts performance when added to a *Fin10* system. Fortunately, the inclusion of importance weights (+*Weights*) reverses this trend, giving us our best result on each test. We saw no consistent improvement from importance weights on the representation-free system.

To better understand the benefits of importance weights, Table 6 reports detailed scores for the Fro14 test set under the *Base+Reps* feature set, as we vary training scenarios. Results on the other test sets are similar. The *Fin10* system achieves high precision but low recall, while the *CoNLL+Fin10* does the opposite. This is because the CoNLL data is much more entity-dense than the Twitter data, which biases systems trained on CoNLL to return too many entities. By down-weighting the CoNLL data, we reduce this bias and gain 6.8 points of precision at the cost of only 2.6 points of recall.

Figure 1 gives learning curves as we add *Fin10* data to *CoNLL* across several feature sets. There is a steady improvement for all systems as the in-domain data grows, and importance weighting increases the impact of in-domain data even at very low quantities. Though the curves shows no sign of flattening out, note that the x-axis is log-scaled.

We have access to roughly 100 million tweets for unsupervised representation learning. Figure 2 provides a learning curve for our *Reps+Weights* system as we increase the percentage of unlabeled tweets used to train both Brown clusters and word vectors from 1.5% to 100% of that data, doubling the amount with each step. With only 1.5 million tweets, average performance is already very good, and we can see that the benefits of scale are starting to level off after we clear 12.5 million.

Test Set	PER	LOC	ORG
Fin10Dev	71.3	72.4	48.8
Rit11	70.8	61.9	36.9
Fro14	69.4	70.2	42.6

Table 7: F1 for our *All Data+Reps+Weights* system, organized by entity class.

Table 7 reports our best system's performance by entity class. For all three test sets, ORG is the most difficult. ORG is perhaps the broadest entity class, but we suspect it is also the most likely to be annotated inconsistently, as it is rife with subtle distinctions: bands (ORG) versus musicians (PER); companies (ORG) versus their products (O); and teams (ORG) versus their home cities (LOC). During an inspection of 25 incorrect ORG predictions by our best system, drawn from a test on *Fin10Dev*, we found 10 cases where the gold standard was questionable. Two of these incorrectly placed "the" inside a chunk, ("[the Mariners]" is wrong; "the [Mariners]" is right), while the remaining 8 involved company-product distinctions, which are tricky even for human annotators. The NER task is not always as intuitive as we would like, and *organizations* tend to highlight these difficulties.

5.2 Comparison with Linguistic Resources

Thus far, we have restricted ourselves to a setting without access to linguistic resources, but for some

	Base +X	Reps +X
$\emptyset$	42.9	62.4
[P]OS Tags	47.1	63.0
[G]azetteers	52.8	63.2
[P]+[G]	55.6	63.5

Table 8: Adding linguistic resources to our baseline and representation-enabled systems, as measured by F1 averaged over 3 test sets. All systems are trained on CoNLL+Fin10, and all but Base+$\emptyset$ use data weighting.

languages, such as English, rich resources exist and can be very useful. We now examine how word representations compare and interact with gazetteers and POS taggers.

For gazetteers, we use those included with the Illinois NER system (Ratinov and Roth, 2009), generating features that indicate when a word appears as part of a phrase found in a gazetteer. For POS tags, we use the CMU Twitter Tagger (Owoputi et al., 2013), and generate POS tag indicators for the current word and for tags within a 2-word window. For some of our corpora, notably CoNLL and Rit11, the corpus tokenization did not match the POS tagger's tokenization. We resolve mismatches by allowing the POS tagger to further tokenize the input sentence to better match its assumptions. After POS tagging, we merge any split tokens back to the original tokenization, picking a representative tag from among merged tags according to a priority list (verb > noun > adjective, etc.). The POS tags may have performed better if we had used the tagger's native tokenization throughout.[4]

Results of our comparison are shown in Table 8. Comparing *Base*+$\emptyset$ and *Base+[P]+[G]*, we see that linguistic resources boost the baseline's performance considerably. Turning to *Reps+[P]+[G]*, we see that adding word representations to linguistic resources provides another substantial boost of 7.9 F1. Conversely, adding linguistic resources to a system that already has representations increases F1 by only 1.1 points, indicating that not much new information is being added. The per-feature analysis indicates that much of this boost comes from the gazetteers.

System	Rit11	Fro14
PHMS14 Baseline	77.4	82.1
PHMS14 Dict $\prec$ Web	78.5	83.9
All Data+Reps+Weights	82.3	86.4
All Data+Ling+Reps+Weights	82.6	86.9

Table 9: Comparison with the state-of-the-art, reporting test F1. Both the gold-standard and the system outputs have `@user` names deterministically tagged as PER.

5.3 Comparison with the State-of-the-Art

In Table 9, we compare our best system, including linguistic resources, to the state-of-the-art results reported by Plank et al. (2014).[5] In order to create a fair comparison, we post-process both our system output and the gold-standard to tag all @user names as PER, just as they do.

Like our system, their baseline includes CoNLL and Twitter data, and uses Brown clusters trained on a comparable number of unlabeled tweets. Their strongest system uses distant supervision over linked web-pages to create artificial training data. But we are able to outperform it with our vector representations and importance weights. Note that this comparison is not perfect, as they train on a much larger pool of crowd-sourced, NER-annotated tweets, consisting of 170k tokens compared to our 17k. The size of their training data is balanced by the fact that its annotations were automatically correctly using MACE (Hovy et al., 2013), where ours were corrected manually, making it unclear which group has the advantage. Nonetheless, our results establish a new state-of-the-art for both test sets, and they do so using only 1k annotated tweets.

6 Analysis

We inspected 100 tweets from the Rit11 test set, focusing on the output from our primary system, *Base+Reps+Weights*, and our baseline, *Base*, both trained on the *CoNLL+Fin10* data. We noted cases where the primary system improved upon the baseline, and cases where it failed to achieve the gold-standard, and placed the phenomena we observed into bins. In general, the baseline was observed

[4] Inconsistent tokenization also hinders the word representations, which were constructed from a corpus tokenized by the CMU Twokenizer.

[5] We omit the Fin10 test set from this comparison, as Plank et al. (2014) test on the entirety of Fin10, while we have divided it into training and development sets.

(*a*) RT @USER : Christmas:PER was so much better when there was a santa :(#allteensthings
RT @USER : Christmas was so much better when there was a santa :(#allteensthings

(*b*) Lmao . I have a feeling Imma:ORG get yelled at tomorrow . Big time . XD Ehh oh well
Lmao . I have a feeling Imma get yelled at tomorrow . Big time . XD Ehh oh well

(*c*) I pray an give God glory even when im in pain , hurting , or crying .
I pray an give God:PER glory even when im in pain , hurting , or crying .

(*d*) Anyone know what days/times that you can smoke hookah at the mix (cma center) in corbin:PER .
Anyone know what days/times that you can smoke hookah at the mix (cma center) in corbin:LOC .

Figure 3: (a,b): examples where the baseline (top) is improved by our final system (bottom)
(c,d): examples where our final system (top) falls short of the gold-standard (bottom)

to rely heavily on local context and capitalization, while the primary system has a much stronger global prior on a given type's entity assignment.

Reps+Weights improved the baseline in 54 out of 100 tweets. There were 31 cases where the primary system corrected a baseline error caused by a misleading capitalization cue. Some of these, such as Figure 3(a) are patched by world knowledge provided by word representations, but many simply reflect a reduced reliance on capitalization. We were surprised to find only 11 cases where Twitter's informal language led to an error, often due to a vaguely name-shaped colloquialism, such as in 3(b). 6 of these 11 cases were fixed by the primary system.

Reps+Weights fell short of the gold-standard in 62 of 100 tweets. We observed 39 recall errors that were difficult to divide into smaller bins. These entities were often missed despite clear capitalization cues, as in Figure 3(c). This particular example is actually a symptom of inconsistent annotation: CoNLL and Rit11 consistently annotate *God* as a person, while our Fin10 training data leaves *God* untagged. The next largest class of errors consists of 11 problems caused by uniform casing (all caps or all lowercase). We also have 5 remaining errors due to informal language, which are interesting, as they highlight gaps in our representations. These include cases where the system generates false entities for variants of rare words (*Tidying* → *Tidyin*), or unusual lengthenings (*Yayaayayay*, as opposed to the well-attested *Yayayayay*). We also saw cases where entities were missed due to creative punctuation (*Go V-I-K-I-N-G-S!*). Finally, we found 4 cases where the system actually over-relies on its word representations, such as in 3(d), where the global PER interpretation of *corbin* overrides a fairly strong LOC signal provided by the local context word *in*.

7 Discussion

We have shown that the combination of Brown clusters, word vectors, and a simple data weighting scheme is sufficient to establish a new state-of-the-art on two Twitter NER test sets, using only 1,000 annotated tweets. We have designed our experiments to emphasize the dramatic impact of word representations in this domain, and to clarify the effects of in- and out-of-domain training sets.

Word representations learned on a large, unlabeled Twitter corpus have addressed a surprising number of issues with inconsistent capitalization and informal language. However, our continuing problems with uncased tweets and unusual colloquialisms demonstrate that there are still many human-readable words that remain a mystery to our system. In response to these observations, we would like to investigate more flexible representations, perhaps similar to those of Botha and Blunsom (2014), who use a linear combination of morpheme vectors to create representations that can generalize across words with similar forms.

Acknowledgments

Thanks to the anonymous reviewers for their helpful comments, to Alan Ritter for providing us with our corpus of unlabeled tweets, and to Barbara Plank for providing us with our three labeled corpora.

References

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

Phil Blunsom and Trevor Cohn. 2011. A hierarchical pitman-yor process hmm for unsupervised part of speech induction. In *ACL*, pages 865–874, Portland, Oregon, USA, June.

Jan A. Botha and Phil Blunsom. 2014. Compositional Morphology for Word Representations and Language Modelling. In *ICML*, Beijing, China.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

Razvan C Bunescu and Raymond J Mooney. 2005. A shortest path dependency kernel for relation extraction. In *EMNLP*, pages 724–731.

M. Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *EMNLP*.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*.

Berry de Bruijn, Colin Cherry, Svetlana Kiritchenko, Joel Martin, and Xiaodan Zhu. 2011. Machine-learned solutions for three stages of clinical information extraction: the state of the art at i2b2 2010. *Journal of the American Medical Informatics Association*, 18(5):557–562.

Tim Finin, Will Murnane, Anand Karandikar, Nicholas Keller, Justin Martineau, and Mark Dredze. 2010. Annotating named entities in twitter data with crowdsourcing. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 80–88.

Hege Fromreide, Dirk Hovy, and Anders Søgaard. 2014. Crowdsourcing and annotating NER for Twitter #drift. In *LREC*, pages 2544–2547, Reykjavik, Iceland.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with mace. In *HLT-NAACL*, pages 1120–1130, Atlanta, Georgia, June.

Nikos Karampatziakis and John Langford. 2010. Online importance weight aware updates. *arXiv preprint arXiv:1011.1576*.

Percy Liang. 2005. *Semi-supervised learning for natural language*. Ph.D. thesis, Massachusetts Institute of Technology.

Dekang Lin and Xiaoyun Wu. 2009. Phrase clustering for discriminative learning. In *Proceedings of the Joint Conference of the ACL and the AFNLP*, pages 1030–1038, Singapore, August.

Xiaohua Liu, Shaodian Zhang, Furu Wei, and Ming Zhou. 2011. Recognizing named entities in tweets. In *ACL*, pages 359–367, Portland, Oregon, USA, June.

André F. T. Martins, Kevin Gimpel, Noah A. Smith, Eric P. Xing, Pedro M. Q. Aguiar, and Mário A. T. Figueiredo. 2010. Learning structured classifiers with dual coordinate descent. Technical Report CMU-ML-10-109, Carnegie Mellon University.

Andrew McCallum and Wei Li. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *CoNLL*, pages 188–191. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR Workshop*.

Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In *HLT-NAACL*, pages 337–342.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *HLT-NAACL*, pages 380–390.

Alexandre Passos, Vineet Kumar, and Andrew McCallum. 2014. Lexicon infused phrase embeddings for named entity resolution. In *CoNLL*, pages 78–86.

Barbara Plank, Dirk Hovy, Ryan McDonald, and Anders Søgaard. 2014. Adapting taggers to Twitter with not-so-distant supervision. In *COLING*, pages 1783–1792, Dublin, Ireland.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *CoNLL*, pages 147–155.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *ACL*, pages 1375–1384.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *EMNLP*, pages 133–142.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *EMNLP*, pages 1524–1534, Edinburgh, Scotland, UK.

Alan Ritter, Oren Etzioni, Sam Clark, et al. 2012. Open domain event extraction from twitter. In *KDD*, pages 1104–1112.

Sunita Sarawagi and William W Cohen. 2004. Semi-markov conditional random fields for information extraction. In *NIPS*, pages 1185–1192.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *CoNLL*, pages 142–147.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *ACL*, pages 384–394.

Peter D Turney. 2012. Domain and function: A dual-space model of semantic relations and compositions. *Journal of Artificial Intelligence Research*, pages 533–585.