Improved Part-of-Speech Tagging for Online Conversational Text with Word Clusters

Olutobi Owoputi* **Brendan O'Connor*** **Chris Dyer***
Kevin Gimpel† **Nathan Schneider*** **Noah A. Smith***
*School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA
†Toyota Technological Institute at Chicago, Chicago, IL 60637, USA
Corresponding author: brenocon@cs.cmu.edu

Abstract

We consider the problem of part-of-speech tagging for informal, online conversational text. We systematically evaluate the use of large-scale unsupervised word clustering and new lexical features to improve tagging accuracy. With these features, our system achieves state-of-the-art tagging results on both Twitter and IRC POS tagging tasks; Twitter tagging is improved from 90% to 93% accuracy (more than 3% absolute). Qualitative analysis of these word clusters yields insights about NLP and linguistic phenomena in this genre. Additionally, we contribute the first POS annotation guidelines for such text and release a new dataset of English language tweets annotated using these guidelines. Tagging software, annotation guidelines, and large-scale word clusters are available at: **http://www.ark.cs.cmu.edu/TweetNLP** This paper describes release 0.3 of the "CMU Twitter Part-of-Speech Tagger" and annotated data.

1 Introduction

Online conversational text, typified by microblogs, chat, and text messages,[1] is a challenge for natural language processing. Unlike the highly edited genres that conventional NLP tools have been developed for, conversational text contains many nonstandard lexical items and syntactic patterns. These are the result of unintentional errors, dialectal variation, conversational ellipsis, topic diversity, and creative use of language and orthography (Eisenstein, 2013). An example is shown in Fig. 1. As a result of this widespread variation, standard modeling assumptions that depend on lexical, syntactic, and orthographic regularity are inappropriate. There is preliminary work on social media part-of-speech (POS) tagging (Gimpel et al., 2011), named entity recognition (Ritter et al., 2011; Liu et al., 2011), and parsing (Foster et al., 2011), but accuracy rates are still significantly lower than traditional well-edited genres like newswire. Even web text parsing, which is a comparatively easier genre than social media, lags behind newspaper text (Petrov and McDonald, 2012), as does speech transcript parsing (McClosky et al., 2010).

ikr	smh	he	asked	fir	yo	last
!	G	O	V	P	D	A
name	so	he	can	add	u	on
N	P	O	V	V	O	P
fb	lololol					
^	!					

Figure 1: Automatically tagged tweet showing nonstandard orthography, capitalization, and abbreviation. Ignoring the interjections and abbreviations, it glosses as *He asked for your last name so he can add you on Facebook.* The tagset is defined in Appendix A. Refer to Fig. 2 for word clusters corresponding to some of these words.

To tackle the challenge of novel words and constructions, we create a new Twitter part-of-speech tagger—building on previous work by Gimpel et al. (2011)—that includes new large-scale distributional features. This leads to state-of-the-art results in POS tagging for both Twitter and Internet Relay Chat (IRC) text. We also annotated a new dataset of tweets with POS tags, improved the annotations in the previous dataset from Gimpel et al., and developed annotation guidelines for manual POS tagging of tweets. We release all of these resources to the research community:

- an open-source part-of-speech tagger for online conversational text (§2);
- unsupervised Twitter word clusters (§3);
- an improved emoticon detector for conversational text (§4);

[1] Also referred to as *computer-mediated communication*.

- POS annotation guidelines (§5.1); and
- a new dataset of 547 manually POS-annotated tweets (§5).

2 MEMM Tagger

Our tagging model is a first-order maximum entropy Markov model (MEMM), a discriminative sequence model for which training and decoding are extremely efficient (Ratnaparkhi, 1996; McCallum et al., 2000).[2] The probability of a tag y_t is conditioned on the input sequence $\boldsymbol{x}$ and the tag to its left y_{t-1}, and is parameterized by a multiclass logistic regression:

$$p(y_t = k \mid y_{t-1}, \boldsymbol{x}, t; \boldsymbol{\beta}) \propto \exp\left(\beta^{(trans)}_{y_{t-1},k} + \textstyle\sum_j \beta^{(obs)}_{j,k} f_j(\boldsymbol{x}, t)\right)$$

We use transition features for every pair of labels, and extract base observation features from token t and neighboring tokens, and conjoin them against all $K = 25$ possible outputs in our coarse tagset (Appendix A). Our feature sets will be discussed below in detail.

Decoding. For experiments reported in this paper, we use the $O(|\boldsymbol{x}|K^2)$ Viterbi algorithm for prediction; K is the number of tags. This exactly maximizes $p(\boldsymbol{y} \mid \boldsymbol{x})$, but the MEMM also naturally allows a faster $O(|\boldsymbol{x}|K)$ left-to-right greedy decoding:

for $t = 1 \ldots |\boldsymbol{x}|$:
  $\hat{y}_t \leftarrow \arg\max_k p(y_t = k \mid \hat{y}_{t-1}, \boldsymbol{x}, t; \boldsymbol{\beta})$

which we find is 3 times faster and yields similar accuracy as Viterbi (an insignificant accuracy decrease of less than 0.1% absolute on the DAILY547 test set discussed below). Speed is paramount for social media analysis applications—which often require the processing of millions to billions of messages—so we make greedy decoding the default in the released software.

[2] Although when compared to CRFs, MEMMs theoretically suffer from the "label bias" problem (Lafferty et al., 2001), our system substantially outperforms the CRF-based taggers of previous work; and when comparing to Gimpel et al. system with similar feature sets, we observed little difference in accuracy. This is consistent with conventional wisdom that the quality of lexical features is much more important than the parametric form of the sequence model, at least in our setting: part-of-speech tagging with a small labeled training set.

This greedy tagger runs at 800 tweets/sec. (10,000 tokens/sec.) on a single CPU core, about 40 times faster than Gimpel et al.'s system. The tokenizer by itself (§4) runs at 3,500 tweets/sec.[3]

Training and regularization. During training, the MEMM log-likelihood for a tagged tweet $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ is the sum over the observed token tags y_t, each conditional on the tweet being tagged and the observed previous tag (with a start symbol before the first token in $\boldsymbol{x}$),

$$\ell(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\beta}) = \textstyle\sum_{t=1}^{|\boldsymbol{x}|} \log p(y_t \mid y_{t-1}, \boldsymbol{x}, t; \boldsymbol{\beta}).$$

We optimize the parameters $\boldsymbol{\beta}$ with OWL-QN, an L_1-capable variant of L-BFGS (Andrew and Gao, 2007; Liu and Nocedal, 1989) to minimize the regularized objective

$$\arg\min_{\boldsymbol{\beta}} \; -\tfrac{1}{N} \textstyle\sum_{\langle \boldsymbol{x}, \boldsymbol{y} \rangle} \ell(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\beta}) + R(\boldsymbol{\beta})$$

where N is the number of tokens in the corpus and the sum ranges over all tagged tweets $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ in the training data. We use elastic net regularization (Zou and Hastie, 2005), which is a linear combination of L_1 and L_2 penalties; here j indexes over all features:

$$R(\boldsymbol{\beta}) = \lambda_1 \textstyle\sum_j |\beta_j| + \frac{1}{2}\lambda_2 \sum_j \beta_j^2$$

Using even a very small L_1 penalty eliminates many irrelevant or noisy features.

3 Unsupervised Word Clusters

Our POS tagger can make use of any number of possibly overlapping features. While we have only a small amount of hand-labeled data for training, we also have access to billions of tokens of *unlabeled* conversational text from the web. Previous work has shown that unlabeled text can be used to induce unsupervised word clusters which can improve the performance of many supervised NLP tasks (Koo et al., 2008; Turian et al., 2010; Täckström et al., 2012, *inter alia*). We use a similar approach here to improve tagging performance for online conversational text. We also make our induced clusters publicly available in the hope that they will be useful for other NLP tasks in this genre.

[3] Runtimes observed on an Intel Core i5 2.4 GHz laptop.

	Binary path	Top words (by frequency)
A1	111010100010	lmao lmfao lmaoo lmaooo hahahahaha lool ctfu rofl loool lmfaoo lmfaooo lmaoooo lmbo **lolololol**
A2	111010100011	haha hahaha hehe hahahaha hahah aha hehehe ahaha hah hahahah kk hahaa ahah
A3	111010100100	yes yep yup nope yess yesss yessss ofcourse yeap likewise yepp yesh yw yuup yus
A4	111010100101	yeah yea nah naw yeahh nooo yeh noo noooo yeaa **ikr** nvm yeahhh nahh nooooo
A5	111010111011100	**smh** jk #fail #random #fact smfh #smh #winning #realtalk smdh #dead #justsaying
B	011101011	**u** yu yuh yhu uu yuu yew y0u yuhh youh yhuu iget yoy yooh yuo 😗 yue juu ʊ dya youz yyou
C	11100101111001	w fo fa fr fro ov fer **fir** whit abou aft serie fore fah fuh w/her w/that fron isn agains
D	111101011000	facebook **fb** itunes myspace skype ebay tumblr bbm flickr aim msn netflix pandora
E1	0011001	tryna gon finna bouta trynna boutta gne fina gonn tryina fenna qone trynaa qon
E2	0011000	gonna gunna gona gna guna gnna ganna qonna gonnna gana qunna gonne goona
F	0110110111	soo sooo soooo sooooo soooooo sooooooo soooooooo sooooooooo soooooooooo
G1	11101011001010	;) :p :-) xd ;-) ;d (; :3 ;p =p :-p =)) ;] xdd #gno xddd >:) ;-p >:d 8-) ;-d
G2	11101011001011	:) (: =) :)) :] ☺ :') =] ^_^ :))) ^.^ [: ;)) 😊 ((: ^__^ (= ^-^ :))))
G3	1110101100111	:(:/ -_- -.- :-(:'(d: :| :s -__- =(=/ >.< -___- :-/ </3 :\ -____- ;(/: :((>_< =[:[#fml
G4	111010110001	<3 ♥ xoxo <33 xo <333 ❤ ♡ #love s2 <URL-twitition.com> #neversaynever <3333

Figure 2: Example word clusters (HMM classes): we list the most probable words, starting with the most probable, in descending order. Boldfaced words appear in the example tweet (Figure 1). The binary strings are root-to-leaf paths through the binary cluster tree. For example usage, see e.g. `search.twitter.com`, `bing.com/social` and `urbandictionary.com`.

3.1 Clustering Method

We obtained hierarchical word clusters via Brown clustering (Brown et al., 1992) on a large set of unlabeled tweets.[4] The algorithm partitions words into a base set of 1,000 clusters, and induces a hierarchy among those 1,000 clusters with a series of greedy agglomerative merges that heuristically optimize the likelihood of a hidden Markov model with a one-class-per-lexical-type constraint. Not only does Brown clustering produce effective features for discriminative models, but its variants are better unsupervised POS taggers than some models developed nearly 20 years later; see comparisons in Blunsom and Cohn (2011). The algorithm is attractive for our purposes since it scales to large amounts of data.

When training on tweets drawn from a single day, we observed time-specific biases (e.g., numerical dates appearing in the same cluster as the word *tonight*), so we assembled our unlabeled data from a random sample of 100,000 tweets per day from September 10, 2008 to August 14, 2012, and filtered out non-English tweets (about 60% of the sample) using `langid.py` (Lui and Baldwin, 2012).[5] Each tweet was processed with our tokenizer and lowercased. We normalized all at-mentions to ⟨*@MENTION*⟩ and URLs/email addresses to their domains (e.g. `http://bit.ly/dP8rR8` ⇒ ⟨*URL-bit.ly*⟩). In an effort to reduce spam, we removed duplicated tweet texts (this also removes retweets) before word clustering. This normalization and cleaning resulted in 56 million unique tweets (847 million tokens). We set the clustering software's count threshold to only cluster words appearing 40 or more times, yielding 216,856 word types, which took 42 hours to cluster on a single CPU.

3.2 Cluster Examples

Fig. 2 shows example clusters. Some of the challenging words in the example tweet (Fig. 1) are highlighted. The term *lolololol* (an extension of *lol* for "laughing out loud") is grouped with a large number of laughter acronyms (A1: "laughing my (fucking) ass off," "cracking the fuck up"). Since expressions of laughter are so prevalent on Twitter, the algorithm creates another laughter cluster (A1's sibling A2), that tends to have onomatopoeic, non-acronym variants (e.g., *haha*). The acronym *ikr* ("I know, right?") is grouped with expressive variations of "yes" and "no" (A4). Note that A1–A4 are grouped in a fairly specific subtree; and indeed, in this message *ikr* and

[4] As implemented by Liang (2005), v. 1.3: `https://github.com/percyliang/brown-cluster`

[5] `https://github.com/saffsd/langid.py`

lololol are both tagged as interjections.

smh ("shaking my head," indicating disapproval) seems related, though is always tagged in the annotated data as a miscellaneous abbreviation (G); the difference between acronyms that are interjections versus other acronyms may be complicated. Here, *smh* is in a related but distinct subtree from the above expressions (A5); its usage in this example is slightly different from its more common usage, which it shares with the other words in its cluster: message-ending expressions of commentary or emotional reaction, sometimes as a metacomment on the author's message; e.g., *Maybe you could get a guy to date you if you actually respected yourself #smh* or *There is really NO reason why other girls should send my boyfriend a goodmorning text #justsaying*.

We observe many variants of categories traditionally considered closed-class, including pronouns (B: *u* = "you") and prepositions (C: *fir* = "for").

There is also evidence of grammatical categories specific to conversational genres of English; clusters E1–E2 demonstrate variations of single-word contractions for "going to" and "trying to," some of which have more complicated semantics.[6]

Finally, the HMM learns about orthographic variants, even though it treats all words as opaque symbols; cluster F consists almost entirely of variants of "so," their frequencies monotonically decreasing in the number of vowel repetitions—a phenomenon called "expressive lengthening" or "affective lengthening" (Brody and Diakopoulos, 2011; Schnoebelen, 2012). This suggests a future direction to jointly model class sequence and orthographic information (Clark, 2003; Smith and Eisner, 2005; Blunsom and Cohn, 2011).

We have built an HTML viewer to browse these and numerous other interesting examples.[7]

3.3 Emoticons and Emoji

We use the term *emoticon* to mean a face or icon constructed with traditional alphabetic or punctuation symbols, and *emoji* to mean symbols rendered in software as small pictures, in line with the text.

Since our tokenizer is careful to preserve emoticons and other symbols (see §4), they are clustered just like other words. Similar emoticons are clustered together (G1–G4), including separate clusters of happy [[*:) =)* ^_^]], sad/disappointed [[*:/ :(* -_- *</3*]], love [[♡*xoxo* ♡.♡]] and winking [[*;) (^_-)*]] emoticons. The clusters are not perfectly aligned with our POS annotation guidelines; for example, the "sad" emoticon cluster included emotion-bearing terms that our guidelines define as non-emoticons, such as *#ugh*, *#tear*, and *#fml* ("fuck my life"), though these seem potentially useful for sentiment analysis.

One difficult task is classifying different types of symbols in tweets: our annotation guidelines differentiate between emoticons, punctuation, and garbage (apparently non-meaningful symbols or tokenization errors). Several Unicode character ranges are reserved for emoji-style symbols (including the three Unicode hearts in G4); however, depending on the user's software, characters in these ranges might be rendered differently or not at all. We have found instances where the clustering algorithm groups proprietary iOS emoji symbols along with normal emoticons; for example, the character U+E056, which is interpreted on iOS as a smiling face, is in the same G2 cluster as smiley face emoticons. The symbol U+E12F, which represents a picture of a bag of money, is grouped with the words *cash* and *money*.

3.4 Cluster-Based Features

Since Brown clusters are hierarchical in a binary tree, each word is associated with a tree path represented as a bitstring with length ≤ 16; we use prefixes of the bitstring as features (for all prefix lengths $\in \{2, 4, 6, \ldots, 16\}$). This allows sharing of statistical strength between similar clusters. Using prefix features of hierarchical clusters in this way was similarly found to be effective for named-entity recognition (Turian et al., 2010) and Twitter POS tagging (Ritter et al., 2011).

When checking to see if a word is associated with a cluster, the tagger first normalizes the word using the same techniques as described in §3.1, then creates a priority list of fuzzy match transformations

[6] One coauthor, a native speaker of the Texan English dialect, notes "finna" (short for "fixing to", cluster E1) may be an immediate future auxiliary, indicating an immediate future tense that is present in many languages (though not in standard English). To illustrate: "She finna go" approximately means "She will go," but sooner, in the sense of "She is about to go."

[7] `http://www.ark.cs.cmu.edu/TweetNLP/cluster_viewer.html`

of the word by removing repeated punctuation and repeated characters. If the normalized word is not in a cluster, the tagger considers the fuzzy matches. Although only about 3% of the tokens in the development set (§6) did not appear in a clustering, this method resulted in a relative error decrease of 18% among such word tokens.

3.5 Other Lexical Features

Besides unsupervised word clusters, there are two other sets of features that contain generalized lexical class information. We use the tag dictionary feature from Gimpel et al., which adds a feature for a word's most frequent part-of-speech tag.[8] This can be viewed as a feature-based domain adaptation method, since it gives lexical type-level information for standard English words, which the model learns to map between PTB tags to the desired output tags. Second, since the lack of consistent capitalization conventions on Twitter makes it especially difficult to recognize names—Gimpel et al. and Foster et al. (2011) found relatively low accuracy on proper nouns—we added a token-level name list feature, which fires on (non-function) words from names from several sources: Freebase lists of celebrities and video games (Google, 2012), the Moby Words list of US Locations,[9] and lists of male, female, family, and proper names from Mark Kantrowitz's name corpus.[10]

4 Tokenization and Emoticon Detection

Word segmentation on Twitter is challenging due to the lack of orthographic conventions; in particular, punctuation, emoticons, URLs, and other symbols may have no whitespace separation from textual words (e.g. *no:-d,yes* should parse as four tokens), and internally may contain alphanumeric symbols that could be mistaken for words: a naive *split(/[^a-zA-Z0-9]+/)* tokenizer thinks the words "p" and "d" are among the top 100 most common words on Twitter, due to misanalysis of *:p* and *:d*. Traditional Penn Treebank–style tokenizers are hardly better, often breaking a string of punctuation characters into a single token per character.

We rewrote `twokenize` (O'Connor et al., 2010), a rule-based tokenizer, emoticon, and URL detector, for use in the tagger. Emoticons are especially challenging, since they are open-class and productive. We revise O'Connor et al.'s regular expression grammar that describes possible emoticons, adding a grammar of horizontal emoticons (e.g. -_-), known as "Eastern-style,"[11] though we observe high usage in English-speaking Twitter (Fig. 2, G2–G3). We also add a number of other improvements to the patterns. Because this system was used as preprocessing for the word clustering experiment in §3, we were able to infer the emoticon clusters in Fig. 2. Furthermore, whether a token matches the emoticon pattern is also used as a feature in the tagger (§2).

URL recognition is also difficult, since the *http://* is often dropped, resulting in protocol-less URLs like *about.me*. We add recognition patterns for these by using a list of top-level and country domains.

5 Annotated Dataset

Gimpel et al. (2011) provided a dataset of POS-tagged tweets consisting almost entirely of tweets sampled from one particular day (October 27, 2010). We were concerned about overfitting to time-specific phenomena; for example, a substantial fraction of the messages are about a basketball game happening that day.

We created a new test set of 547 tweets for evaluation. The test set consists of one random English tweet from every day between January 1, 2011 and June 30, 2012. In order for a tweet to be considered English, it had to contain at least one English word other than a URL, emoticon, or at-mention. We noticed biases in the outputs of `langid.py`, so we instead selected these messages completely manu-

[8] Frequencies came from the *Wall Street Journal* and Brown corpus sections of the Penn Treebank. If a word has multiple PTB tags, each tag is a feature with value for the frequency rank; e.g. for three different tags in the PTB, this feature gives a value of 1 for the most frequent tag, 2/3 for the second, etc. Coarse versions of the PTB tags are used (Petrov et al., 2011). While 88% of words in the dictionary have only one tag, using rank information seemed to give a small but consistent gain over only using the most common tag, or using binary features conjoined with rank as in Gimpel et al.

[9] `http://icon.shef.ac.uk/Moby/mwords.html`

[10] `http://www.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/nlp/corpora/names/0.html`

[11] `http://en.wikipedia.org/wiki/List_of_emoticons`

ally (going through a random sample of one day's messages until an English message was found).

5.1 Annotation Methodology

Gimpel et al. provided a tagset for Twitter (shown in Appendix A), which we used unmodified. The original annotation guidelines were not published, but in this work we recorded the rules governing tagging decisions and made further revisions while annotating the new data.[12] Some of our guidelines reiterate or modify rules made by Penn Treebank annotators, while others treat specific phenomena found on Twitter (refer to the next section).

Our tweets were annotated by two annotators who attempted to match the choices made in Gimpel et al.'s dataset. The annotators also consulted the POS annotations in the Penn Treebank (Marcus et al., 1993) as an additional reference. Differences were reconciled by a third annotator in discussion with all annotators.[13] During this process, an inconsistency was found in Gimpel et al.'s data, which we corrected (concerning the tagging of *this*/*that*, a change to 100 labels, 0.4%). The new version of Gimpel et al.'s data (called OCT27), as well as the newer messages (called DAILY547), are both included in our data release.

5.2 Compounds in Penn Treebank vs. Twitter

Ritter et al. (2011) annotated tweets using an augmented version of the PTB tagset and presumably followed the PTB annotation guidelines. We wrote new guidelines because the PTB conventions are inappropriate for Twitter in several ways, as shown in the design of Gimpel et al.'s tagset. Importantly, "compound" tags (e.g., nominal+verbal and nominal+possessive) are used because tokenization is difficult or seemingly impossible for the nonstandard word forms that are commonplace in conversational text.

For example, the PTB tokenization splits contractions containing apostrophes: *I'm* ⇒ *I*/PRP *'m*/VBP. But conversational text often contains variants that resist a single PTB tag (like *im*), or even challenge traditional English grammatical categories (like *imma* or *umma*, which both mean "I am going to"). One strategy would be to analyze these forms into a PTB-style tokenization, as discussed in Forsyth (2007), who proposes to analyze *doncha* as *do*/VBP *ncha*/PRP, but notes it would be difficult. We think this is impossible to handle in the rule-based framework used by English tokenizers, given the huge (and possibly growing) number of large compounds like *imma*, *gonna*, *w/that*, etc. These are not rare: the word clustering algorithm discovers hundreds of such words as statistically coherent classes (e.g. clusters E1 and E2 in Fig. 2); and the word *imma* is the 962nd most common word in our unlabeled corpus, more frequent than *cat* or *near*.

We do not attempt to do Twitter "normalization" into traditional written English (Han and Baldwin, 2011), which we view as a lossy translation task. In fact, many of Twitter's unique linguistic phenomena are due not only to its informal nature, but also a set of authors that heavily skews towards younger ages and minorities, with heavy usage of dialects that are different than the standard American English most often seen in NLP datasets (Eisenstein, 2013; Eisenstein et al., 2011). For example, we suspect that *imma* may implicate tense and aspect markers from African-American Vernacular English.[14] Trying to impose PTB-style tokenization on Twitter is linguistically inappropriate: should the lexico-syntactic behavior of casual conversational chatter by young minorities be straightjacketed into the stylistic conventions of the 1980s *Wall Street Journal*? Instead, we would like to directly analyze the syntax of online conversational text on its own terms.

Thus, we choose to leave these word forms untokenized and use compound tags, viewing compositional multiword analysis as challenging future work.[15] We believe that our strategy is sufficient for many applications, such as chunking or named entity recognition; many applications such as sentiment analysis (Turney, 2002; Pang and Lee, 2008, §4.2.3), open information extraction (Carlson et al., 2010; Fader et al., 2011), and information retrieval (Allan and Raghavan, 2002) use POS

[12] The annotation guidelines are available online at `http://www.ark.cs.cmu.edu/TweetNLP/`

[13] Annotators are coauthors of this paper.

[14] See "Tense and aspect" examples in `http://en.wikipedia.org/wiki/African_American_Vernacular_English`

[15] For example, *wtf* has compositional behavior in "Wtf just happened??", but only debatably so in "Huh wtf".

	#Msg.	#Tok.	Tagset	Dates
OCT27	1,827	26,594	App. A	Oct 27-28, 2010
DAILY547	547	7,707	App. A	Jan 2011–Jun 2012
NPSCHAT	10,578	44,997	PTB-like	Oct–Nov 2006
(w/o sys. msg.)	7,935	37,081		
RITTERTW	789	15,185	PTB-like	unknown

Table 1: Annotated datasets: number of messages, tokens, tagset, and date range. More information in §5, §6.3, and §6.2.

patterns that seem quite compatible with our approach. More complex downstream processing like parsing is an interesting challenge, since contraction parsing on traditional text is probably a benefit to current parsers. We believe that any PTB-trained tool requires substantial retraining and adaptation for Twitter due to the huge genre and stylistic differences (Foster et al., 2011); thus tokenization conventions are a relatively minor concern. Our simple-to-annotate conventions make it easier to produce new training data.

6 Experiments

We are primarily concerned with performance on our annotated datasets described in §5 (OCT27, DAILY547), though for comparison to previous work we also test on other corpora (RITTERTW in §6.2, NPSCHAT in §6.3). The annotated datasets are listed in Table 1.

6.1 Main Experiments

We use OCT27 to refer to the entire dataset described in Gimpel et al.; it is split into training, development, and test portions (OCT27TRAIN, OCT27DEV, OCT27TEST). We use DAILY547 as an additional test set. Neither OCT27TEST nor DAILY547 were extensively evaluated against until final ablation testing when writing this paper.

The total number of features is 3.7 million, all of which are used under pure L_2 regularization; but only 60,000 are selected by elastic net regularization with $(\lambda_1, \lambda_2) = (0.25, 2)$, which achieves nearly the same (but no better) accuracy as pure L_2,[16] and we use it for all experiments. We observed that it was

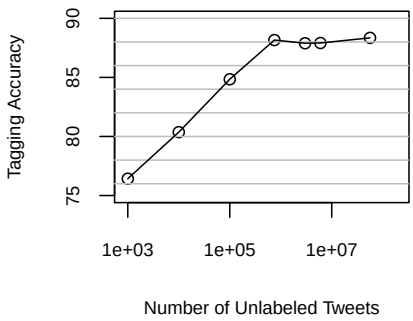

Figure 3: OCT27 development set accuracy using only clusters as features.

Model	In dict.	Out of dict.
Full	93.4	85.0
No clusters	92.0 (−1.4)	79.3 (−5.7)
Total tokens	*4,808*	*1,394*

Table 3: DAILY547 accuracies (%) for tokens in and out of a traditional dictionary, for models reported in rows 1 and 3 of Table 2.

possible to get radically smaller models with only a slight degradation in performance: $(4, 0.06)$ has 0.5% worse accuracy but uses only 1,632 features, a small enough number to browse through manually.

First, we evaluate on the new test set, training on all of OCT27. Due to DAILY547's statistical representativeness, we believe this gives the best view of the tagger's accuracy on English Twitter text. The full tagger attains **93.2%** accuracy (final row of Table 2).

To facilitate comparisons with previous work, we ran a series of experiments training only on OCT27's training and development sets, then report test results on both OCT27TEST and all of DAILY547, shown in Table 2. Our tagger achieves substantially higher accuracy than Gimpel et al. (2011).[17]

Feature ablation. A number of ablation tests indicate the word clusters are a very strong source of lexical knowledge. When dropping the tag dictionaries and name lists, the word clusters maintain most of the accuracy (row 2). If we drop the clusters and rely only on tag dictionaries and namelists, accuracy decreases significantly (row 3). In fact, we can remove *all* observation features except for word clusters—no word features, orthographic fea-

[16] We conducted a grid search for the regularizer values on part of DAILY547, and many regularizer values give the best or nearly the best results. We suspect a different setup would have yielded similar results.

[17] These numbers differ slightly from those reported by Gimpel et al., due to the corrections we made to the OCT27 data, noted in Section 5.1. We retrained and evaluated their tagger (version 0.2) on our corrected dataset.

Feature set	OCT27TEST	DAILY547	NPSCHATTEST	
All features	91.60	92.80	91.19	1
with clusters; without tagdicts, namelists	91.15	92.38	90.66	2
without clusters; with tagdicts, namelists	89.81	90.81	90.00	3
only clusters (and transitions)	89.50	90.54	89.55	4
without clusters, tagdicts, namelists	86.86	88.30	88.26	5
Gimpel et al. (2011) version 0.2	88.89	89.17		6
Inter-annotator agreement (Gimpel et al., 2011)	92.2			7
Model trained on all OCT27		93.2		8

Table 2: Tagging accuracies (%) in ablation experiments. OCT27TEST and DAILY547 95% confidence intervals are roughly $\pm 0.7\%$. Our final tagger uses all features and also trains on OCT27TEST, achieving 93.2% on DAILY547.

tures, affix n-grams, capitalization, emoticon patterns, etc.—and the accuracy is in fact still better than the previous work (row 4).[18]

We also wanted to know whether to keep the tag dictionary and name list features, but the splits reported in Fig. 2 did not show statistically significant differences; so to better discriminate between ablations, we created a lopsided train/test split of all data with a much larger test portion (26,974 tokens), having greater statistical power (tighter confidence intervals of $\pm$ 0.3%).[19] The full system got 90.8% while the no–tag dictionary, no-namelists ablation had 90.0%, a statistically significant difference. Therefore we retain these features.

Compared to the tagger in Gimpel et al., most of our feature changes are in the new lexical features described in §3.5.[20] We do not reuse the other lexical features from the previous work, including a phonetic normalizer (Metaphone), a name list consisting of words that are frequently capitalized, and distributional features trained on a much smaller unlabeled corpus; they are all worse than our new lexical features described here. (We did include, however, a variant of the tag dictionary feature that uses phonetic normalization for lookup; it seemed to yield a small improvement.)

Non-traditional words. The word clusters are especially helpful with words that do not appear in traditional dictionaries. We constructed a dictionary by lowercasing the union of the *ispell* 'American', 'British', and 'English' dictionaries, plus the standard Unix *words* file from Webster's Second International dictionary, totalling 260,985 word types. After excluding tokens defined by the gold standard as punctuation, URLs, at-mentions, or emoticons,[21] 22% of DAILY547's tokens do not appear in this dictionary. Without clusters, they are very difficult to classify (only 79.2% accuracy), but adding clusters generates a 5.7 point improvement—much larger than the effect on in-dictionary tokens (Table 3).

Varying the amount of unlabeled data. A tagger that only uses word clusters achieves an accuracy of 88.6% on the OCT27 development set.[22] We created several clusterings with different numbers of unlabeled tweets, keeping the number of clusters constant at 800. As shown in Fig. 3, there was initially a logarithmic relationship between number of tweets and accuracy, but accuracy (and lexical coverage) levels out after 750,000 tweets. We use the largest clustering (56 million tweets and 1,000 clusters) as the default for the released tagger.

6.2 Evaluation on RITTERTW

Ritter et al. (2011) annotated a corpus of 787 tweets[23] with a single annotator, using the PTB

[18] Furthermore, when evaluating the clusters as *unsupervised* (hard) POS tags, we obtain a many-to-one accuracy of 89.2% on DAILY547. Before computing this, we lowercased the text to match the clusters and removed tokens tagged as URLs and at-mentions.

[19] Reported confidence intervals in this paper are 95% binomial normal approximation intervals for the proportion of correctly tagged tokens: $\pm 1.96\sqrt{p(1-p)/n_{tokens}} \lesssim 1/\sqrt{n}$.

[20] Details on the exact feature set are available in a technical report (Owoputi et al., 2012), also available on the website.

[21] We retain hashtags since by our guidelines a #-prefixed token is ambiguous between a hashtag and a normal word, e.g. *#1* or *going #home*.

[22] The only observation features are the word clusters of a token and its immediate neighbors.

[23] `https://github.com/aritter/twitter_nlp/blob/master/data/annotated/pos.txt`

Tagger	Accuracy
This work	90.0 ± 0.5
Ritter et al. (2011), basic CRF tagger	85.3
Ritter et al. (2011), trained on more data	88.3

Table 4: Accuracy comparison on Ritter et al.'s Twitter POS corpus (§6.2).

Tagger	Accuracy
This work	93.4 ± 0.3
Forsyth (2007)	90.8

Table 5: Accuracy comparison on Forsyth's NPSCHAT IRC POS corpus (§6.3).

tagset plus several Twitter-specific tags, referred to in Table 1 as RITTERTW. Linguistic concerns notwithstanding (§5.2), for a controlled comparison, we train and test our system on this data with the same 4-fold cross-validation setup they used, attaining 90.0% (±0.5%) accuracy. Ritter et al.'s CRF-based tagger had 85.3% accuracy, and their best tagger, trained on a concatenation of PTB, IRC, and Twitter, achieved 88.3% (Table 4).

6.3 IRC: Evaluation on NPSCHAT

IRC is another medium of online conversational text, with similar emoticons, misspellings, abbreviations and acronyms as Twitter data. We evaluate our tagger on the NPS Chat Corpus (Forsyth and Martell, 2007),[24] a PTB-part-of-speech annotated dataset of Internet Relay Chat (IRC) room messages from 2006.

First, we compare to a tagger in the same setup as experiments on this data in Forsyth (2007), training on 90% of the data and testing on 10%; we average results across 10-fold cross-validation.[25] The full tagger model achieved 93.4% (±0.3%) accuracy, significantly improving over the best result they report, 90.8% accuracy with a tagger trained on a mix of several POS-annotated corpora.

We also perform the ablation experiments on this corpus, with a slightly different experimental setup: we first filter out system messages then split data into 5,067 training and 2,868 test messages. Results show a similar pattern as the Twitter data (see final column of Table 2). Thus the Twitter word clusters are also useful for language in the medium of text chat rooms; we suspect these clusters will be applicable for deeper syntactic and semantic analysis in other online conversational text mediums, such as text messages and instant messages.

[24] Release 1.0: `http://faculty.nps.edu/cmartell/NPSChat.htm`

[25] Forsyth actually used 30 different 90/10 random splits; we prefer cross-validation because the same test data is never repeated, thus allowing straightforward confidence estimation of accuracy from the number of tokens (via binomial sample variance, footnote 19). In all cases, the models are trained on the same amount of data (90%).

7 Conclusion

We have constructed a state-of-the-art part-of-speech tagger for the online conversational text genres of Twitter and IRC, and have publicly released our new evaluation data, annotation guidelines, open-source tagger, and word clusters at `http://www.ark.cs.cmu.edu/TweetNLP`.

Acknowledgements

This research was supported in part by the National Science Foundation (IIS-0915187 and IIS-1054319).

A Part-of-Speech Tagset

N	common noun
O	pronoun (personal/WH; not possessive)
^	proper noun
S	nominal + possessive
Z	proper noun + possessive
V	verb including copula, auxiliaries
L	nominal + verbal (e.g. *i'm*), verbal + nominal (*let's*)
M	proper noun + verbal
A	adjective
R	adverb
!	interjection
D	determiner
P	pre- or postposition, or subordinating conjunction
&	coordinating conjunction
T	verb particle
X	existential *there*, predeterminers
Y	X + verbal
#	hashtag (indicates topic/category for tweet)
@	at-mention (indicates a user as a recipient of a tweet)
~	discourse marker, indications of continuation across multiple tweets
U	URL or email address
E	emoticon
$	numeral
,	punctuation
G	other abbreviations, foreign words, possessive endings, symbols, garbage

Table 6: POS tagset from Gimpel et al. (2011) used in this paper, and described further in the released annotation guidelines.

References

J. Allan and H. Raghavan. 2002. Using part-of-speech patterns to reduce query ambiguity. In *Proc. of SIGIR*.

G. Andrew and J. Gao. 2007. Scalable training of L_1-regularized log-linear models. In *Proc. of ICML*.

P. Blunsom and T. Cohn. 2011. A hierarchical Pitman-Yor process HMM for unsupervised part of speech induction. In *Proc. of ACL*.

S. Brody and N. Diakopoulos. 2011. Cooooooooooooooollllllllllllll!!!!!!!!!!!!!!: using word lengthening to detect sentiment in microblogs. In *Proc. of EMNLP*.

P. F. Brown, P. V. de Souza, R. L. Mercer, V. J. Della Pietra, and J. C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4).

A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka Jr, and T. M. Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proc. of AAAI*.

A. Clark. 2003. Combining distributional and morphological information for part of speech induction. In *Proc. of EACL*.

J. Eisenstein, N. A. Smith, and E. P. Xing. 2011. Discovering sociolinguistic associations with structured sparsity. In *Proc. of ACL*.

J. Eisenstein. 2013. What to do about bad language on the internet. In *Proc. of NAACL*.

A. Fader, S. Soderland, and O. Etzioni. 2011. Identifying relations for open information extraction. In *Proc. of EMNLP*.

E. N. Forsyth and C. H. Martell. 2007. Lexical and discourse analysis of online chat dialog. In *Proc. of ICSC*.

E. N. Forsyth. 2007. Improving automated lexical and discourse analysis of online chat dialog. Master's thesis, Naval Postgraduate School.

J. Foster, O. Cetinoglu, J. Wagner, J. L. Roux, S. Hogan, J. Nivre, D. Hogan, and J. van Genabith. 2011. #hardtoparse: POS tagging and parsing the Twitterverse. In *Proc. of AAAI-11 Workshop on Analysing Microtext*.

K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith. 2011. Part-of-speech tagging for Twitter: Annotation, features, and experiments. In *Proc. of ACL*.

Google. 2012. Freebase data dumps. `http://download.freebase.com/datadumps/`.

B. Han and T. Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a #twitter. In *Proc. of ACL*.

T. Koo, X. Carreras, and M. Collins. 2008. Simple semi-supervised dependency parsing. In *Proc. of ACL*.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML*.

P. Liang. 2005. Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology.

D. C. Liu and J. Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1).

X. Liu, S. Zhang, F. Wei, and M. Zhou. 2011. Recognizing named entities in tweets. In *Proc. of ACL*.

M. Lui and T. Baldwin. 2012. langid.py: An off-the-shelf language identification tool. In *Proc. of ACL*.

M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2).

A. McCallum, D. Freitag, and F. Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. In *Proc. of ICML*.

D. McClosky, E. Charniak, and M. Johnson. 2010. Automatic domain adaptation for parsing. In *Proc. of NAACL*.

B. O'Connor, M. Krieger, and D. Ahn. 2010. TweetMotif: exploratory search and topic summarization for Twitter. In *Proc. of AAAI Conference on Weblogs and Social Media*.

O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, and N. Schneider. 2012. Part-of-speech tagging for Twitter: Word clusters and other advances. Technical Report CMU-ML-12-107, Carnegie Mellon University.

B. Pang and L. Lee. 2008. *Opinion mining and sentiment analysis*. Now Publishers.

S. Petrov and R. McDonald. 2012. Overview of the 2012 shared task on parsing the web. Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL).

S. Petrov, D. Das, and R. McDonald. 2011. A universal part-of-speech tagset. *arXiv preprint arXiv:1104.2086*.

A. Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proc. of EMNLP*.

A. Ritter, S. Clark, Mausam, and O. Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proc. of EMNLP*.

T. Schnoebelen. 2012. Do you smile with your nose? Stylistic variation in Twitter emoticons. *University of Pennsylvania Working Papers in Linguistics*, 18(2):14.

N. A. Smith and J. Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proc. of ACL*.

O. Täckström, R. McDonald, and J. Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proc. of NAACL*.

J. Turian, L. Ratinov, and Y. Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proc. of ACL*.

P. D. Turney. 2002. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proc. of ACL*.

H. Zou and T. Hastie. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320.