Tune Your Brown Clustering, Please

Leon Derczynski
University of Sheffield
leon@dcs.shef.ac.uk

Sean Chester
Aarhus University
schester@cs.au.dk

Kenneth S. Bøgh
Aarhus University
ksb@cs.au.dk

Abstract

Brown clustering, an unsupervised hierarchical clustering technique based on n-gram mutual information, has proven useful in many NLP applications. However, most uses of Brown clustering employ the same default configuration; the appropriateness of this configuration has gone predominantly unexplored. Accordingly, we present information for practitioners on the behaviour of Brown clustering in order to assist hyper-parametre tuning, in the form of a theoretical model of Brown clustering utility. This model is then evaluated empirically in two sequence labelling tasks over two text types. We explore the dynamic between the input corpus size, chosen number of classes, and quality of the resulting clusters, which has an impact for any approach using Brown clustering. In every scenario that we examine, our results reveal that the values most commonly used for the clustering are sub-optimal.

1 Introduction

Brown clustering (Brown et al., 1992) uses distributional information to group similar words. Unsupervised, it induces a hierarchical clustering over words to form a binary tree (e.g. Figure 1). This hierarchical clustering has recently been used in thousands of computational linguistics papers, often for feature generation. However, no work exists describing the behaviour and hyper-parametre tuning effects of Brown clustering; even the original paper concentrates on implementation rather than its behaviour.

Except for a few forays off the beaten track (e.g. Christodoulopoulos et al. (2010), Owoputi et al. (2012), Derczynski et al. (2015a)), default parametres dominate; either 800 or 1000 Brown clusters are generated in nearly every published use. Few experiments use other configurations, and we are not aware of any prior work on hyper-parametre tuning for Brown clustering.

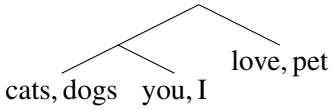

Figure 1: A binary, hierarchical clustering of semantically similar entries. Each leaf corresponds to a cluster of words (i.e., a "class") and leaves near to their common ancestors correspond to clusters that are similar to each other.

This paper addresses this information gap, providing practitioners with principled insights into the algorithm. We provide an analysis of how Brown clustering adds information over input, and, based on this, describe models for the effect that corpus size and cluster count have on the quality of results. These models are then tested in two sequence labeling tasks, cf. Qu et al. (2015). Finally, we compare the initial analysis to observations, leading to concrete advice for practitioners.

2 Background

Brown clustering uses mutual information to determine distributional similarity, placing similar words in the same cluster and similar clusters nearby in the binary tree. This is an unsupervised learned representation of language from the input corpus (Bengio et al., 2013). In the main implementation of Brown clustering (Liang, 2005), mutual information is measured at the bigram level. The resulting structure of word types can be used as feature representations in many NLP tasks, leading to quick, solid performance increases (Turian et al., 2010). In fact, as well as producing effective discriminative features, unsupervised hierarchical clusterings like Brown often lead to better taggers than models developed 20 years later (Blunsom and Cohn (2011), Owoputi et al. (2013)).

Bit path	Word types
00111001	can cn cann caan cannn ckan shalll ccan caaan cannnn caaaan
001011111001	ii id ion iv ll iii ud wd uma ul idnt provoking hed 1+1 ididnt hast ine 2+2 idw #thingsblackpeopledo iiii #onlywhitepeople dost doan uon apt-get

Table 1: Sample Brown clusters over English tweets.[1] Each set of terms is a leaf in the hierarchy.

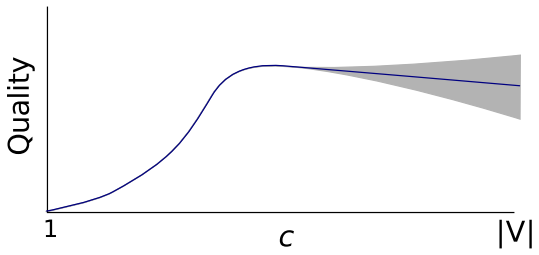

Figure 2: Expected cluster quality as c increases, given a hypothetical ideal cluster quality function.

In practice, Brown clustering takes an input corpus T and number of classes c, and uses mutual information to assign each term in the corpus vocabulary V to one of the c classes. Ideally, each class contains highly semantically-related words, by virtue of words being distributed according to their meaning (Wittgenstein, 1953). Each class is a leaf on an unbalanced binary tree. The path from the root to each leaf can be described as a bit string, where the i'th bit is 0 iff the path branches left at depth i (e.g., *you,I* is on the path `01` in Figure 1). Brown clustering posits that leaves with longer common path prefices are more semantically related. For example, in Figure 1, the *cats,dog* and *you,I* classes are more similar than either is to the *love,pet* class.

3 A Model for Brown Clustering

Here we outline our model for the behaviour of Brown clustering under various situations. Our goal is to describe how the number of classes, c, affects the quality of the resulting clustering.

Initial values for c might not be appropriate for a given task or data set. Large values of c risk forcing similar words into different classes, underrepresenting their similarity. Conversely, a small c may cluster too coarsely, thereby reducing the discriminative power of resulting representations.

Brown clustering adds two forms of information: *the agglomeration* of terms into similar groups and *the hierarchy* connecting semantically similar groups of terms. At extreme values of c, little is added: if $c = |V|$, each word has its own class and only the hierarchy is added; if $c = 1$, one cluster contains all terms and information is gained from neither clustering nor a hierarchy. So, the information added by clustering increases with $c > 1$, peaks, and then declines towards $|V|$.

However, the information added solely by the hierarchy increases with c and peaks when every word type has its own cluster, i.e., when $c = |V|$, as this gives the maximum number to the tree; we cannot add more leaves than there are word types (given a single root).

Also, a too-small c may produce classes of unequal quality. Table 1 lists two classes derived in Owoputi et al. (2012), with $c = 1000$, on a large social media corpus. The first cluster agglomerates a set of semantically close lexemes, but the second cluster is internally semantically disparate, conflating many different concepts. This could indicate an inadequate value for c that forces many concepts into a too-confined number of classes.

A c exceeding the number of word types is also problematic: each word type should have only one class. This can arise in small datasets and when the vocabulary is particularly formalised (e.g., in a controlled natural language) (Wyner et al., 2010). Indeed, the size of the input dataset not only affects the number of eventual word types (Montemurro, 2001), but also quality of the classes.

For a fixed task and corpus genre, we hypothesise that each corpus size has an optimal c and each c has an optimal corpus size. When increasing a corpus size, new word types and further distributional information is revealed. The new distributional information leads to better-informed assignment of terms to classes, thereby improving the cluster quality. Eventually, however, the profusion of word types outgrows c and semantically dissimilar words will be placed in the same class. Overall, we expect clustering performance to scale as shown in Figure 2: quality increases with c to an optimal value, then dips slightly and levels off with some stochastic variance.

In fact, this behaviour has been observed (but not explained nor analysed) before. Owoputi et al. (2012) comment on the performance of a PoS tagger that for *"different amounts of unlabeled tweets, keeping the number of clusters constant at* 800 [...] *initially there was a logarithmic rela-*

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

tion between number of tweets and accuracy, but from from 750 *thousand to* 56 *million tweets, the tagging accuracy remained relatively constant."*

4 Method

Datasets We evaluate Brown tuning using two text types. For newswire, we use the Reuters RCV1 dataset (Rose et al., 2002). For social media, we draw randomly from a 10% sampling of tweets collected from 2009–2015, filtered for English using *langid.py* (Lui and Baldwin, 2012).

Preprocessing Drawing upon previous work (Turian et al., 2009; Owoputi et al., 2012), input data is preprocessed:

- Newswire data is *cleaned* per Liang (2005);
- Tabs, newlines and carriage returns are replaced with spaces;
- URLs are replaced with a <*URL*> marker;
- @Mentions are replaced by <*Mention*>; and
- Social media data has end-of-sequence markers <*EOS*> between tweets (see below).

Social media text was tokenised using the twokenize tool (O'Connor et al., 2010); newswire, with the Stanford tokenizer (Manning et al., 2014). The *cleaning* is the removal of any sentence where less than 90% of the characters are lowercase letters (excluding whitespace). This was not applied to tweets, as non-alphabet characters are markedly more frequent in social media text and an equivalent threshold is unclear. Cleaning has a notable effect on the RCV1 dataset, which has much potentially misleading non-text data such as numeric tables. Ultimately, $|T| = 1\,008.6$c for 72.1M social media tweets. For newswire, $|T| = 114.8$M.

Terminals We note that Brown et al. (1992) assume a corpus long enough ($T \rightarrow \infty$) that the final term in Equation 1 tends to 1, and so $Pr(c_1|c_2)$ tends to the relative frequency of consecutive classes c_1c_2.

$$Pr(c_1|c_2) = \frac{C(c_1c_2)}{T} \times \frac{C(c_1)}{\sum_c C(c_1c)} \quad (1)$$

When corpora are composed of long, structured documents, bigrams are unlikely to cross the boundaries of unrelated sentences. However, in social media corpora there is little running discourse: each document is $\leq$ 140 characters and usually just one sentence. Running discourse only occurs when consecutive messages are from the same user and temporally ordered (or perhaps relate to a hashtag or conversation, which may be non-linear). Given the uniformity of Twitter sampling (Kergl et al., 2014), this continuity is unlikely. Therefore, we introduce an <*EOS*> marker after each tweet to break bigrams. This also captures some sentence position information.

5 Evaluation

The effect of class count (c) and corpus size (number of tokens, $|T|$) is measured extrinsically in two scenarios. Firstly, the generated clusters are used as a plug-in to the CMU Twitter part-of-speech tagger, replacing the supplied clusters and paths. This evaluation only covers social media. Secondly, the clusters are used to support feature generation in named entity recognition. This covers newswire and social media. The scenarios and corresponding evaluation measures are described below. Clusters are generated from all word types, even those that occur only once in the corpus.

Note critically that we aim to observe the performance sensitivity to input parametres, and to gain insights for tuning Brown clustering. Achieving new top scores in any task is *not* the goal.

5.1 Part of Speech Tagging

Owoputi et al. (2013) present a PoS tagger for tweets which relies on (among other features) Brown clusters. A reference clustering (and two evaluation datasets) is provided with the tagger, which we substitute with newly generated clusters. To observe the impact of tuning Brown clustering, we vary input parametres to produce new clusters and measure the tagger's resultant tagging accuracy at token level. The "oct27" training and test splits are used.

5.2 Named Entity Recognition

We simplify NER to isolate the impact of c and $|T|$. A CRF (Okazaki, 2007) is used to train and classify NER models. The only features are Brown cluster path prefices of length [4,6,10,20] for newswire, as per Ratinov and Roth (2009), and [2,4,8,16] for newswire, as per Plank et al. (2014).

For newswire, we train and evaluate on the CoNLL data (Tjong Kim Sang and De Meulder, 2003) taking RCV clusters as input. For social media, we use the CRF with passive-aggressive updates to overcome some social media noise (Derczynski and Bontcheva, 2014), and train and eval-

	$T = 1M$	$T = 8$M		$T = 62.5$k
c	**SM F1**	**SM F1**	**NW F1**	**NW F1**
10	19.5	19.5	12.1	16.73
20	19.5	19.5	16.0	16.65
40	19.5	19.5	18.3	17.51
80	19.8	20.6	24.7	22.19
160	21.6	28.7	34.2	23.71
320	23.5	31.9	38.3	26.10
640	34.2	40.7	42.1	28.36
1000	*37.0*	*48.4*	*43.0*	*29.84*
1280	34.9	48.5	44.2	30.51
2560	41.2	49.2	44.5	31.57
5120	37.7	51.1	46.1	33.20
9229		-	-	33.23
10240	37.8	47.3	45.8	n/a

Table 2: NER accuracy, varying the number of classes c and corpus size $|T|$. For $T = 62.5$k, $|V| = 9\,229$.

uate on the Ritter et al. (2011) data, converted to PER / LOC / ORG / MISC and using the splits given by Derczynski et al. (2015b).

Additionally, we investigate feature representations. As we know that Brown clustering adds two kinds of information – the grouping of word types into classes and the hierarchy between classes (Section 3) – we isolate these two and analyse their individual performance. We evaluate performance of class-only and path-only features over the RCV1 data, due to its larger evaluation partition. Path-only features are extracted by truncating at $[1 : bits - 2]$, e.g., the cluster path 1100101 yields features (1,11,110,1100,11001).

6 Analysis

As expected, extrinsic performance increases as number of classes c rises for a given corpus size $|T|$, and also as $|T|$ rises for a given c, supporting our hypothesis that performance improves as c grows from 1. As c continues rising, word types are distributed more thinly across classes. Results show that performance levels off, and even begins to decrease (Table 2). In this experiment, we used an 8M token corpus and up to 10240 classes.

While this shows the effect of cluster quality decreasing when there are both too many and too few clusters, it does not approach the extreme value of c where there is one class for each word type. Thus, we ran another experiment varying cluster size but on a smaller corpus, which allowed examination of performance nearer to $c = |V|$. For this, we took 62500 tokens of cleaned RCV1, which contained 9229 word types, and kept the same range of c values. The news genre (NW) was selected for two reasons: the larger evaluation set provides better resolution in results, and the reduced lexical variation means lower word type proliferation, giving more distributional information for the same data. Results are given in Table 2. The plateauing behaviour matches the predicted idealised performance curve in Figure 2 reasonably well. Note that the NER extrinsic evaluation relies more on hierarchical information than clustering, and so the drop in quality may be less pronounced than in other tasks.

T	**NW F1**	**SM F1**	T	**NW F1**	**SM F1**
8K	21.5	23.5	2M	39.1	38.6
16K	24.4	24.8	4M	41.4	45.0
32K	28.5	26.2	8M	43.0	48.4
62.5K	29.9	27.5	16M	44.2	50.2
125K	30.6	25.9	32M	45.6	54.2
250K	31.8	31.2	64M	46.9	51.7
500K	35.5	34.9	125M	n/a	51.7
1M	36.5	37.0	250M	n/a	53.6

Table 3: NER accuracy, varying corpus size $|T|$; $c = 1000$.

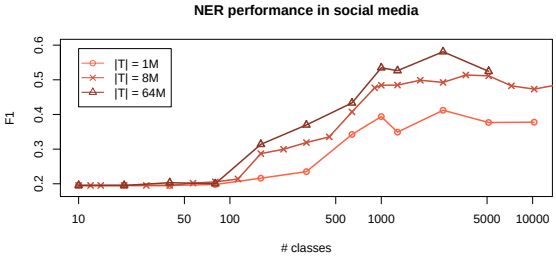

Figure 3: Social media NER F1.

For the social media data (SM), we observe unstable quality for large $|T|$ (Table 3). This shows the point where too much data has been added and the classes have become noisy. Additional data for some $|T|$ values is shown in Figure 3. As the noise is balanced by the addition of distributional information, we do not expect cluster quality to plummet rapidly, but rather hover; the data reflects this.

For PoS tagging, we see that there is a peak performance with $c = 640$, after which accuracy drops unstably (Table 4). This matches our expectations. In fact, the performance for $c = 1000$ (the value used to generate the original clusters for the CMU tagger) is a local minimum in our test. The PoS task involves a lot of other factors, and so is not as close an estimate of clustering quality as the NER task is, but it does make use of both the clusters and the hierarchy. No clear result came from varying $|T|$ with a fixed c (Table 5), unlike in NER, where increasing $|T|$ had a strong impact.

Some low values of c are particularly bad, espe-

Classes c	Oct27 TA	Classes c	Oct27 TA
10	62.9	640	80.0
20	66.3	*1 000*	*76.9*
40	66.3	1 280	78.2
80	76.7	2 560	75.0
160	76.4	5 120	76.5
320	72.3	10 240	79.4

Table 4: PoS token accuracy (TA), varying c (8M tokens).

# tokens (T)	Oct27 TA	# tokens (T)	Oct27 TA
8K	78.7	2M	77.1
16K	80.6	4M	79.1
32K	76.9	8M	76.9
62.5K	79.5	16M	68.7
125K	79.5	32M	74.6
250K	76.4	64M	77.0
500K	76.1	125M	73.8
1M	72.6	250M	74.2

Table 5: PoS token accuracy, varying corpus size ($c = 1000$).

cially in the social media NER task, as in Table 2: with 40 or fewer classes, performance was consistently very low. This may be due to the smaller size of the SM evaluation set and high lexical variation in tweets, compared to newswire, where performance is also low but increases (sluggishly). As expected, we see (for SM) that larger input corpora benefit from higher c.[2] The default value gave sub-optimal results in every case.

The separation of cluster-path and class information (Tables 6, 7; Figures 4, 5) was revealing. In both cases, low values of c give not static but worsening performance as $|T|$ rises (see e.g. the low-performance region in the lower back right of Figure 6). This is likely due to the effect c has on determining the number of items considered for a merge at any point; as the input corpus grows, this "window" comprises an ever-decreasing proportion of available word types. Also, performance is more sensitive to increases in c when $|T|$ is large, whereas increases under smaller $|T|$ are milder.

With the class-only experiment, performance peaks and then declines as $c \to |V|$, as expected (Section 3). The extreme class-only case, $c = |V|$, is one class per word, equivalent to a one-hot representation.[3] In the path-only experiment, perfor-

[2] During this we did in fact out-perform the leading system in a large study of Twitter NER systems; performance with $|T| = 32M, m = 1000$ (Table 3) was better than the best overall F1 in Table 3 of Derczynski et al. (2015b), despite using solely Brown cluster features.

[3] We do not use a minimum token frequency cutoff; if one

$c \downarrow$; $\lvert T\rvert$:	8K	16K	32K	250K	1M	8M	64M
10	11.9	11.3	14.0	13.5	12.8	12.7	12.7
40	12.6	14.1	17.4	16.5	15.1	16.8	20.0
80	13.2	14.7	19.2	22.7	21.3	22.3	19.0
160	12.8	13.3	18.9	23.0	28.1	30.4	28.5
320	18.8	14.0	20.6	27.2	33.3	36.0	38.0
640	19.6	16.7	19.2	30.0	32.8	40.3	41.9
1280	18.9	23.4	26.7	31.1	35.8	42.0	45.7
2560	21.7	**26.1**	30.2	31.4	36.8	42.8	47.4
5120	**24.0**	23.0	**31.9**	33.7	39.0	43.6	48.2
10240	-	-	28.1	**37.0**	**40.3**	**45.2**	**49.3**

Table 6: NER accuracy (F1); path-only (nonterminal) features; newswire. Bold indicates best c for a given $|T|$.

$c \downarrow$; $\lvert T\rvert$:	8K	16K	32K	250K	1M	8M	64M
10	12.6	14.1	17.4	15.3	14.7	12.1	11.7
40	12.7	14.2	17.6	16.6	16.9	18.4	22.3
80	13.9	15.5	19.5	23.1	22.7	24.8	29.1
160	14.3	16.8	20.2	25.7	29.3	34.3	32.7
320	16.2	17.7	19.7	28.1	33.9	38.3	40.4
640	18.4	20.2	23.1	30.2	35.6	41.8	46.5
1280	21.5	23.4	26.5	31.7	36.9	**44.1**	46.5
2560	22.2	24.5	28.9	33.4	**39.2**	44.1	**48.2**
5120	**22.2**	**25.1**	30.4	**34.5**	38.5	43.6	46.8
10240	-	-	**30.5**	34.1	37.7	41.7	45.3

Table 7: NER accuracy (F1); class-only feature; newswire. Bold indicates best c for T.

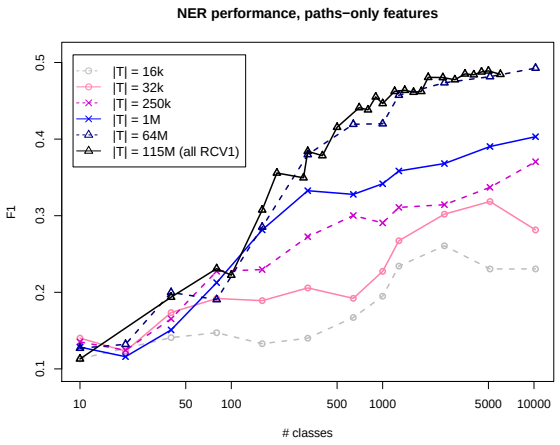

Figure 4: Using decomposed class prefices, without cluster ID, for paths-only features.

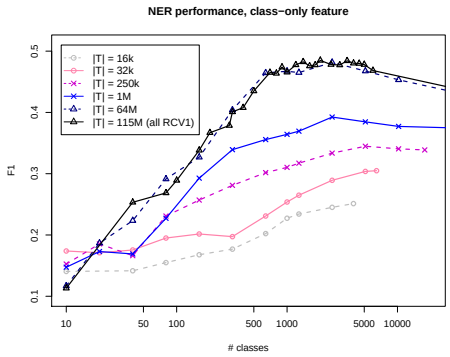

Figure 5: Using Brown class / cluster ID as sole feature. A 3D plot of these data points and others is shown in Figure 6.

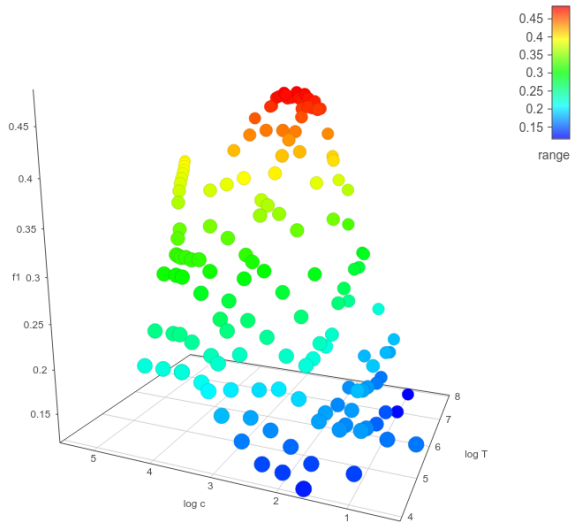

Figure 6: 3D plot of F1 using only cluster information, varying T and c. Interactive version at http://derczynski.com/sheffield/brown-tuning/ .

mance increases with both $|T|$ and c. The advice here is that if $|T|$ is easier to increase than waiting for a large c, then get the big corpus first.

The best possible c behaves oppositely with class-only and path-only information. For class-only, with small corpora, c should be high (or set to $|V|$); as the corpora grow, so the best c levels off (Table 7). Conversely, for path-only, small corpora benefit from lower c, whereas larger corpora do better with high values of c (Table 6). This is because as $c \to |V|$, more path information is added, whereas clustering information decreases, as suggested in Section 3.

To exploit high values of c when $|T|$ is substantial, path features are required. Further, it may be more efficient to try a lower c and a larger $|T|$. In scenarios where the clusters are more important than hierarchical information, choosing too high a value for c is both expensive and risky.

Default values of c are unlikely to perform well, and are often even local minima in performance. Note that performance does not increase monotonically with either $|T|$ or c; this is likely due to poor decisions being made by the algorithm based on the information available at the time under those parameters. As a different tree is generated for every different corpus and class count, and these tree vary almost chaotically across text types and corpus sizes, and also as performance depends on how features are extracted, it is unlikely that a universal formula for selecting c exists. Ceteris paribus, it is reasonable to start finding c through random search (Bergstra and Bengio, 2012) beta-weighted against high c to reduce computation costs (Micenková et al., 2015) and against very low c where extrinsic performance is poor; e.g., something like $c \sim B(\alpha = 1.5, \beta = 5)c_{max}$, with c_{max} in the order of 10^{5+}, based on $|T|$ and our results in both text types.

Supplementary to this paper, we provide many clusters and paths for the two common text types investigated, to help researchers start exploring Brown parametre space for their problem for some values of c, thus deferring the initial large computational costs of running this algorithm.

is used, this equivalency no longer applies.

7 Conclusion

As a community, if Brown clustering is to continue its adoption in so many NLP tasks, we need methods to choose appropriate values for its hyper-parametres. We presented our model of how Brown clustering quality changes depending on its input and tuning. This model was supported in an empirical evaluation.

The target number of classes c has an impact on the utility of the classes. The corpus size $|T|$ also has an impact.

Setting c too low clusters too coarsely; setting it too high forces similar words to be split across clusters. Similarly, a preset c will not be optimal for ever-increasing corpus sizes: just adding more data will eventually make no difference or even reduce cluster quality. We therefore strongly recommend *avoiding* the default value of $c = 1000$, and instead finding values which fully activate this powerful hierarchical clustering technique.

Acknowledgments

This research has received funding from the EU's seventh Framework Programme for research, technological development and demonstration under grant agreement No. 611233, PHEME,[4] and from the WALLVIZ project,[5] supported by the Danish Council for Strategic Research, grant 10-092316.

References

Yoshua Bengio, Aaron Courville, and Pierre Vincent. 2013. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

[4] http://www.pheme.eu/
[5] http://wallviz.dk/

James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(1):281–305.

Phil Blunsom and Trevor Cohn. 2011. A hierarchical Pitman-Yor process HMM for unsupervised part of speech induction. In *Proc. ACL*, pages 865–874.

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.

Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. 2010. Two Decades of Unsupervised POS induction: How far have we come? In *Proc. EMNLP*, pages 575–584. ACL.

Leon Derczynski and Kalina Bontcheva. 2014. Passive-aggressive sequence labeling with discriminative post-editing for recognising person entities in tweets. In *Proc. EACL*, volume 2, pages 69–73.

Leon Derczynski, Isabelle Augenstein, and Kalina Bontcheva. 2015a. USFD: Twitter NER with Drift Compensation and Linked Data. In *Proc. W-NUT workshop, ACL*.

Leon Derczynski, Diana Maynard, Giuseppe Rizzo, Marieke van Erp, Genevieve Gorrell, Raphaël Troncy, Johann Petrak, and Kalina Bontcheva. 2015b. Analysis of named entity recognition and linking for tweets. *Information Processing & Management*, 51(2):32–49.

Dennis Kergl, Robert Roedler, and Sebastian Seeber. 2014. On the endogenesis of Twitter's Spritzer and Gardenhose sample streams. In *Proc. ASONAM*, pages 357–364. IEEE.

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

Marco Lui and Timothy Baldwin. 2012. langid.py: An off-the-shelf language identification tool. In *Proc. ACL*, pages 25–30.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proc. ACL*, pages 55–60.

Barbora Micenková, Brian McWilliams, and Ira Assent. 2015. Learning representations for outlier detection on a budget. *arXiv:cs.LG/1507.08104*.

Marcelo A Montemurro. 2001. Beyond the Zipf–Mandelbrot law in quantitative linguistics. *Physica A: Statistical Mechanics and its Applications*, 300(3):567–578.

Brendan O'Connor, Michel Krieger, and David Ahn. 2010. TweetMotif: Exploratory Search and Topic Summarization for Twitter. In *Proc. ICWSM*, pages 384–385.

Naoaki Okazaki. 2007. CRFsuite: a fast implementation of conditional random fields (CRFs). *URL http://www. chokkan. org/software/crfsuite*.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, and Nathan Schneider. 2012. Part-of-speech tagging for Twitter: Word clusters and other advances. Technical Report CMU-ML-12-107, School of Computer Science, Carnegie Mellon University.

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 *Proc. NAACL*, pages 380–390.

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

Lizhen Qu, Gabriela Ferraro, Liyuan Zhou, Weiwei Hou, Nathan Schneider, and Timothy Baldwin. 2015. Big data small data, in domain out-of domain, known word unknown word: The impact of word representation on sequence labelling tasks. In *Proc. CoNLL*.

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

Alan Ritter, Sam Clark, Oren Etzioni, et al. 2011. Named entity recognition in tweets: an experimental study. In *Proc. EMNLP*, pages 1524–1534.

Tony Rose, Mark Stevenson, and Miles Whitehead. 2002. The Reuters Corpus Volume 1 - from yesterday's news to tomorrow's language resources. In *Proc. LREC*, volume 2, pages 827–832.

Karl Stratos, Do-kyum Kim, Michael Collins, and Daniel Hsu. 2014. A spectral algorithm for learning class-based n-gram models of natural language. *Proc. UAI*.

Erik F Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proc. NAACL*, volume 4, pages 142–147.

Joseph Turian, Lev Ratinov, Yoshua Bengio, and Dan Roth. 2009. A preliminary evaluation of word representations for named-entity recognition. In *Proc. NIPS ws. on Grammar Induction, Representation of Language & Language Learning*, pages 1–8.

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

Ludwig Wittgenstein. 1953. *Philosophical Investigations*. Basic Blackwell, London.

Adam Wyner, Krasimir Angelov, Guntis Barzdins, Danica Damljanovic, Brian Davis, Norbert Fuchs, Stefan Hoefler, Ken Jones, Kaarel Kaljurand, Tobias Kuhn, et al. 2010. On controlled natural languages: Properties and prospects. In *Controlled Natural Language*, pages 281–289. Springer.