Scaling Up Word Clustering

Jon Dehdari[1,2] and **Liling Tan**[2] and **Josef van Genabith**[1,2]
[1]DFKI, Saarbrücken, Germany
{jon.dehdari,josef.van_genabith}@dfki.de
[2]University of Saarland, Saarbrücken, Germany
liling.tan@uni-saarland.de

Abstract

Word clusters improve performance in many NLP tasks including training neural network language models, but current increases in datasets are outpacing the ability of word clusterers to handle them. In this paper we present a novel bidirectional, interpolated, refining, and alternating (BIRA) predictive exchange algorithm and introduce ClusterCat, a clusterer based on this algorithm. We show that ClusterCat is 3–85 times faster than four other well-known clusterers, while also improving upon the predictive exchange algorithm's perplexity by up to 18%. Notably, ClusterCat clusters a 2.5 billion token English News Crawl corpus in 3 hours. We also evaluate in a machine translation setting, resulting in shorter training times achieving the same translation quality measured in BLEU scores. ClusterCat is portable and freely available.

1 Introduction

Words can be grouped into equivalence classes to reduce data sparsity and generalize data. Word clusters are useful in many NLP applications. Within machine translation, word classes are used in word alignment (Brown et al., 1993; Och and Ney, 2000), translation models (Koehn and Hoang, 2007; Wuebker et al., 2013), reordering (Cherry, 2013), preordering (Stymne, 2012), SAMT (Zollmann and Vogel, 2011), and OSM (Durrani et al., 2014).

Word clusterings have also found utility in parsing (Koo et al., 2008; Candito and Seddah, 2010; Kong et al., 2014), chunking (Turian et al., 2010), and NER (Miller et al., 2004; Liang, 2005; Turian et al., 2010; Ritter et al., 2011), among many others.

Word clusters also speed up normalization in training neural network and MaxEnt language models, via class-based decomposition (Goodman, 2001b). This reduces the normalization time from $\mathcal{O}(|V|)$ (the vocabulary size) to $\approx \mathcal{O}(\sqrt{|V|})$.

2 Exchange-Based Clustering

The **exchange algorithm** (Kneser and Ney, 1993) uses an unlexicalized (two-sided) model: $P(w_i|w_{i-1}) = P(w_i|c_i)\,P(c_i|c_{i-1})$ where the class c_i of the predicted word w_i is conditioned on the class c_{i-1} of the previous word w_{i-1}. Goodman (2001a) altered this model so that c_i is conditioned directly upon w_{i-1}: $P(w_i|w_{i-1}) = P(w_i|c_i)\,P(c_i|w_{i-1})$. This fractionates the history more, but it greatly speeds up hypothesizing an exchange since the history doesn't change. The resulting partially lexicalized (one-sided) model gives the accompanying **predictive exchange algorithm** (Uszkoreit and Brants, 2008) a time complexity of $\mathcal{O}((B + |V|) \times |C| \times I)$ where B is the number of unique bigrams, $|C|$ is the number of classes, and I is the number of training iterations, usually < 20.

3 ClusterCat

ClusterCat is word clustering software designed to be fast and scalable, while also improving upon the predictive exchange algorithm. We describe in this section improvements in the model, the algorithm, as well as in the implementation.

3.1 Model and Algorithm

We developed a *bidirectional, interpolated, refining, and alternating* (BIRA) predictive exchange algorithm. The goal of BIRA is to produce better clusters by using multiple, changing models to escape local optima. This uses both forward and reversed bigram class models in order to improve cluster quality by evaluating log-likelihood on two different models. Unlike using trigrams, bidirectional bigram models only linearly increase time and memory requirements, and in fact some data structures can be shared. The two directions are interpolated to allow softer integration of these two models: $P(w_i|w_{i-1}, w_{i+1}) \triangleq P(w_i|c_i) \cdot (\lambda P(c_i|w_{i-1}) + (1-\lambda) P(c_i|w_{i+1}))$. Furthermore, the interpolation weight λ for the forward direction alternates to $1-\lambda$ every a iterations i to help escape local optima. The time complexity is $\mathcal{O}(2 \times (B + |V|) \times |C| \times I)$. The original predictive exchange algorithm can be obtained by setting $\lambda = 1$ and $a = 0$.

Cluster refinement improves both cluster quality *and* speed. The vocabulary is initially clustered into $|G|$ sets, where $|G| \ll |C|$, typically 2–10 . This groups words into broad classes, like nouns, verbs, etc. After a few iterations (i) of this, the full partitioning C_f is explored. Clustering G converges very quickly, typically requiring no more than 3 iterations. In contrast to divisive hierarchical clustering and coarse-to-fine methods (Petrov, 2009), after the initial iterations, any word can still move to any cluster—there is no hard constraint that the more refined partitions be subsets of the initial coarser partitions. This gives more flexibility in optimizing on log-likelihood, especially given the noise that naturally arises from coarser clusterings. We explored cluster refinement over more stages than just two, successively increasing the number of clusters. We observed no improvement over the two-stage method described above.

The contributions of each of these, relative to the original predictive exchange algorithm, are shown in Figure 1 . The data and configurations are discussed in more detail in Section 4. The greatest improvement is due to using lambda inversion (`+Rev`), followed by cluster refinement (`+Refine`), then interpolating the bidirectional models (`+BiDi`), with robust improvements by using all three of these—an

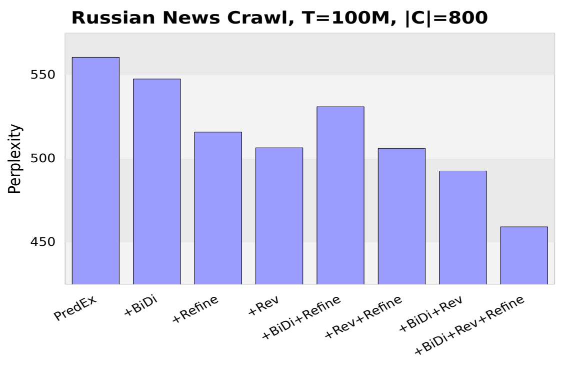

Figure 1: Dev set PP of combinations of improvements to the predictive exchange algorithm (cf. §3.1), using 100M tokens of the Russian News Crawl, with 800 word classes.

18% reduction in perplexity over the predictive exchange algorithm. We have found that both lambda inversion and cluster refinement prevent early convergence at local optima, while bidirectional models give immediate and consistent training set PP improvements, but this is attenuated in a unidirectional evaluation.

3.2 Implementation

We represent the set of bigrams B as an array of records that track the number of predecessors, as well as having a pointer to an array of the predecessors' IDs. This allows for easy prefetching to reduce memory latency, while also keeping memory overhead low. We dispense with the predictive exchange `RemoveWord` procedure for tentative steps, since this does not change the final clustering.

Most of the computation for the predictive exchange algorithm is spent on the logarithm function in $\delta \leftarrow \delta - N(w,c) \cdot \log N(w,c)$.[1] Since the codomain of $N(w,c)$ is $\mathbb{N}_0$, and due to the power law distribution of the algorithm's access to these entropy terms, we precompute $N \cdot \log N$ up to, say 10e+7, with minimal memory requirements.[2] This results in a considerable speedup of around 40% .

4 Experiments

We evaluate ClusterCat on training time, two-sided class-based language model (LM) perplexity

[1] δ is the change in log-likelihood, and $N(w,c)$ is the count of a given word followed by a given class.

[2] This was independently discovered in Botros et al. (2015).

(cf. Brown et al., 1992; Uszkoreit and Brants, 2008), and BLEU scores in phrase-based MT.

4.1 Intrinsic Evaluation

For the two-sided class-based LM task we used 800 and 1200 classes for English, and 800 classes for Russian. The clusterers (cf. Sec. 2) are Brown-cluster (Liang, 2005), ClusterCat (introduced in Section 3), `mkcls` (Och, 1995), Phrasal's clusterer (Green et al., 2014), and word2vec's clustering feature (Mikolov et al., 2013).

The data comes from the 2011–2013 News Crawl monolingual data of the WMT task.[3] For these experiments the data was deduplicated, shuffled, tokenized, digit-conflated, and lowercased. In order to have a large test set, one line per 100 of the resulting corpus was separated into the test set.[4] For English this gave 1B training tokens, 2M training types, and 12M test tokens. For Russian, 550M training tokens, 2.7M training types, and 6M test tokens.

All clusterers had a minimum count threshold of 3 occurrences in the training set. All used 12 threads and 15 iterations, except single-threaded `mkcls` which used the default one iteration. Clusterings were performed on a 2.4 GHz Opteron 8378 machine featuring 16 threads and 64 GB of RAM.

Table 1 presents wall clock times. The predictive exchange-based clusterers (ClusterCat and Phrasal) exhibit slow time growth as $|C|$ increases, while the other three (Brown, `mkcls`, and word2vec) are much more sensitive to $|C|$. ClusterCat is three times faster than Phrasal for all sets. For both English and Russian we observe prohibitive growth for `mkcls`, with the full Russian training set taking over 3 days, compared to 1.5 hours for ClusterCat.

Training Set	Brown	CC	mkcls	Phrasal	w2v
EN, $\|C\| = 800$	12.5	1.4	48.8	5.1	20.6
EN, $\|C\| = 1200$	25.5	1.7	68.8	6.2	33.7
RU, $\|C\| = 800$	14.6	1.5	75.0	5.5	12.0

Table 1: Clustering times (hours) of full training sets. For English, $T = 10^9$; for Russian, $T = 10^{8.74}$.

We performed an additional experiment on ClusterCat, adding more training data.[5] *ClusterCat took 3.0 hours to cluster 2.5 billion training tokens*, using 40 GB of memory for $|C| = 800$. When the number of clusters was tripled to $|C| = 2400$, the same 2.5B corpus was clustered in under 8 hours.

Training Set	Brown	CC	mkcls	Phrasal	w2v
EN, $\|C\| = 800$	160.2	158.1	155.0	178.3	383.4
EN, $\|C\| = 1200$	141.5	140.4	138.4	157.6	330.7
RU, $\|C\| = 800$	350.4	340.7	322.4	389.3	560.9

Table 2: 5-gram two-sided class-based LM PP using 10^9 English training tokens or $10^{8.74}$ Russian training tokens.

The clusterings are also evaluated on the perplexity (PP) of an external 5-gram two-sided class-based LM. Botros et al. (2015) found that the two-sided model (which `mkcls` uses) tends to give better PP in two-sided class-based LM experiments, but the one-sided model of the predictive exchange that we employed produces better PP for training LSTM LMs.

Table 2 shows perplexity results using a varying number of classes. As word2vec is the only clusterer not optimized on log-likelihood, its perplexity is quite high, and remains high as more training data is added.[6] On the other hand, `mkcls` gives the lowest perplexity, although this is an artefact of the two-sided evaluation. ClusterCat gives lower perplexity than the original predictive exchange algorithm (in Phrasal) and Brown clustering. The Russian experiments yielded higher PP for all clusterings, but otherwise the same comparative results. The metaheuristic techniques used in `mkcls` can be applied to other exchange-based clusterers—including ours—for further improvements.

It is also interesting to look at time-sensitive clustering. Figure 2 shows what perplexity can be obtained within a given training time frame. For each clusterer, each successive rightward point in the figure represents an order of magnitude more training data, from 10^6 to 10^9 tokens. *ClusterCat can train on 10 times more data than either `mkcls` or Brown-cluster and produces better perplexity than either, within a given amount of time.*

[3] `http://bit.ly/1SAjeIx`

[4] We provide a script to replicate the data setup at `http://www.dfki.de/~jode03/naacl2016.sh`.

[5] Adding years 2008–2010 and 2014 to the existing English News Crawl training data. This training set was too large for the external class-based LM to fit into memory, so no perplexity evaluation of this clustering was possible.

[6] The skip-gram model within word2vec resulted in even higher PP at almost three times the clustering time, relative to the CBOW model that we used. Using hierarchical softmax with a window of one word on either side gave no appreciable difference in perplexity, while also increasing training time.

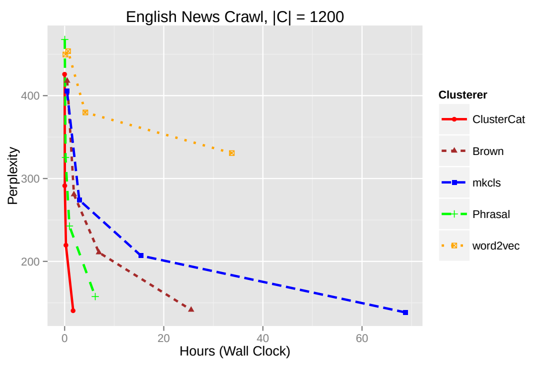

Figure 2: Relationship between two-sided class-based LM PP and the time needed to produce clusters ($|C| = 1200$). Points in the lower-left corner are best.

	$\|C\|$=50	100	200	500	1000
EN-RU	$+0.2^*$	$+0.7$	-0.2	0	$+0.2$
RU-EN	-0.2	-0.1	$+0.1$	$+0.1^*$	$+0.1^{**}$

Table 3: BLEU score changes and significance across varying cluster sizes. Positive values indicate ClusterCat BLEU > `mkcls` BLEU.

4.2 Extrinsic Evaluation

We also evaluated `mkcls` and ClusterCat extrinsically in machine translation, for word alignment. As training sets get larger every year, `mkcls` struggles to keep pace, and is a substantial time bottleneck in MT pipelines. We compare time and BLEU scores of using either `mkcls` or ClusterCat for Russian↔English translation.

The parallel data comes from the WMT-2015 Common Crawl Corpus, News Commentary, Yandex 1M Corpus, and the Wiki Headlines Corpus.[7] The monolingual data consists of 2007–2014 News Commentary and News Crawl articles. The dev and test sets contain 3000 sentences from EN→RU manually translated news articles. We used standard configurations, like truecasing, MGIZA alignment, GDFA phrase extraction, phrase-based Moses, quantized KenLM 5-gram MKN LMs, and MERT tuning.

Table 3 presents the BLEU score changes across varying cluster sizes.[8] The BLEU score differences between using `mkcls` and ClusterCat are minimal but there are a few statistically significant changes, using bootstrap resampling (Koehn, 2004).

[7] `http://bit.ly/1SAjeIx`

[8] *: p-value < 0.05, **: p-value < 0.01 . More results are presented in Dehdari et al. (2016) .

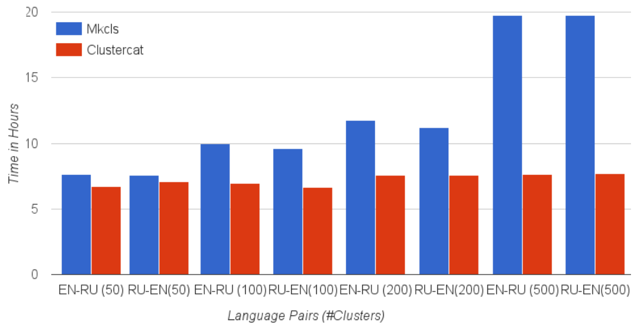

Figure 3: End-to-end translation model training times for English-Russian and Russian-English for various cluster sizes using `mkcls` and ClusterCat.

Figure 3 shows translation model training times, before MERT. Using ClusterCat reduces the translation model training time with 500 clusters from 20 hours using `mkcls` (of which 60% of the time is spent on clustering) to just 8 hours (of which 5% is spent on clustering).

5 Conclusion

In this article we have presented improvements to the predictive exchange algorithm that address longstanding drawbacks of the original algorithm compared to other clustering algorithms. Bidirectional models, lambda inversion, and cluster refinement produce better word clusters, as we showed in several two-sided class-based LM experiments. On these large datasets the quality of the resulting clusters is better than predictive exchange clusters and Brown clusters, and approaches the stochastic exchange clusters produced by `mkcls`, which takes 35–85 times longer.

We also improved upon the speed of the algorithm by cluster refinement and entropy term precalculation. MT experiments showed that word alignment models using ClusterCat fully match those using `mkcls` in BLEU scores, with time savings found by using ClusterCat. The software, as well as additional compatibility and visualization scripts, are available under a Free license at `https://github.com/jonsafari/clustercat`.

Acknowledgements

We would like to thank Hermann Ney, Kazuki Irie, and the reviewers. This work was supported by the QT21 project (Horizon 2020 No. 645452).

References

R. Botros, K. Irie, M. Sundermeyer, and H. Ney. 2015. On Efficient Training of Word Classes and Their Application to Recurrent Neural Network Language Models. In *Proc. INTERSPEECH*, pages 1443–1447.

P. Brown, P. deSouza, R. Mercer, V. Della Pietra, and J. Lai. 1992. Class-Based n-gram Models of Natural Language. *Comp. Ling.*, 18(4):467–479.

P. Brown, S. Della Pietra, V. Della Pietra, and R. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Comp. Ling.*, 19(2):263–311.

M. Candito and D. Seddah. 2010. Parsing Word Clusters. In *Proc. of the Workshop on SPMRL*, pages 76–84.

C. Cherry. 2013. Improved Reordering for Phrase-Based Translation using Sparse Features. In *Proc. NAACL-HLT 2013*, pages 22–31.

J. Dehdari, L. Tan, and J. van Genabith. 2016. BIRA: Improved Predictive Exchange Word Clustering. In *Proc. NAACL*, San Diego, CA, USA.

N. Durrani, P. Koehn, H. Schmid, and A. Fraser. 2014. Investigating the Usefulness of Generalized Word Representations in SMT. In *Proc. of Coling 2014*, pages 421–432.

J. Goodman. 2001a. A Bit of Progress in Language Modeling, Extended Version. Technical Report MSR-TR-2001-72, Microsoft Research.

J. Goodman. 2001b. Classes for Fast Maximum Entropy Training. In *Proc. of ICASSP*, pages 561–564.

S. Green, D. Cer, and C. Manning. 2014. An Empirical Comparison of Features and Tuning for Phrase-based Machine Translation. In *Proc. WMT*, pages 466–476.

R. Kneser and H. Ney. 1993. Improved clustering techniques for class-based statistical language modelling. In *Proc. EUROSPEECH*, pages 973–976.

P. Koehn and H. Hoang. 2007. Factored Translation Models. In *Proc. EMNLP-CoNLL*, pages 868–876.

P. Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proc. EMNLP*, pages 388–395.

L. Kong, N. Schneider, S. Swayamdipta, A. Bhatia, C. Dyer, and N. Smith. 2014. A Dependency Parser for Tweets. In *Proc. EMNLP*, pages 1001–1012.

T. Koo, X. Carreras, and M. Collins. 2008. Simple Semi-supervised Dependency Parsing. In *Proc. ACL: HLT*, pages 595–603.

P. Liang. 2005. Semi-Supervised Learning for Natural Language. Master's thesis, MIT.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *Proc. ICLR*.

S. Miller, J. Guinness, and A. Zamanian. 2004. Name Tagging with Word Clusters and Discriminative Training. In Susan Dumais, Daniel Marcu, and Salim Roukos, editors, *Proc. HLT-NAACL*, pages 337–342.

F. Och and H. Ney. 2000. A Comparison of Alignment Models for Statistical Machine Translation. In *Proc. Coling*, pages 1086–1090.

F. Och. 1995. Maximum-Likelihood-Schätzung von Wortkategorien mit Verfahren der kombinatorischen Optimierung. Bachelor's thesis (Studienarbeit), Universität Erlangen-Nürnburg.

S. Petrov. 2009. *Coarse-to-Fine Natural Language Processing*. Ph.D. thesis, UC-Berkeley.

A. Ritter, S. Clark, Mausam, and O. Etzioni. 2011. Named Entity Recognition in Tweets: An Experimental Study. In *Proc. EMNLP*, pages 1524–1534.

S. Stymne. 2012. Clustered Word Classes for Preordering in Statistical Machine Translation. In *Proc. of the Joint Workshop on Unsupervised and Semi-Supervised Learning in NLP*, pages 28–34.

J. Turian, L. Ratinov, and Y. Bengio. 2010. Word Representations: A Simple and General Method for Semi-Supervised Learning. In *Proc. ACL*, pages 384–394.

J. Uszkoreit and T. Brants. 2008. Distributed Word Clustering for Large Scale Class-Based Language Modeling in Machine Translation. In *Proc. ACL: HLT*, pages 755–762.

J. Wuebker, S. Peitz, F. Rietig, and H. Ney. 2013. Improving Statistical Machine Translation with Word Class Models. In *Proc. EMNLP*, pages 1377–1381.

A. Zollmann and S. Vogel. 2011. A Word-Class Approach to Labeling PSCFG Rules for Machine Translation. In *Proc. ACL-HLT*, pages 1–11.