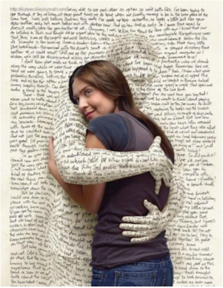
Words

and their abstractions

Jon Dehdari

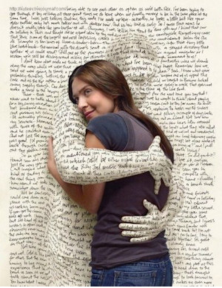
November 24, 2015

Too Many Words!



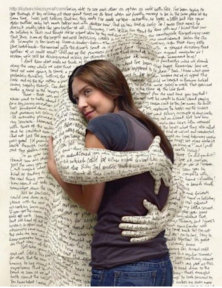
- Languages have too many words for statistical models of language

Too Many Words!



- Languages have too many words for statistical models of language
- We need some way to generalize them

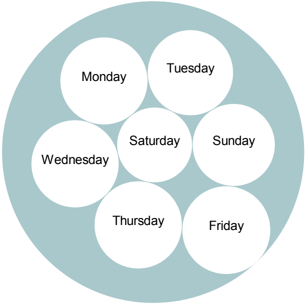
Too Many Words!

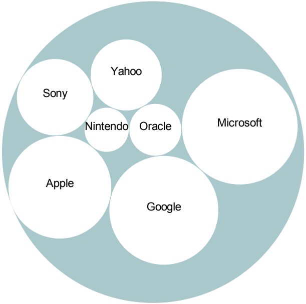


- Languages have too many words for statistical models of language
- We need some way to generalize them
- Let's treat some words like other words

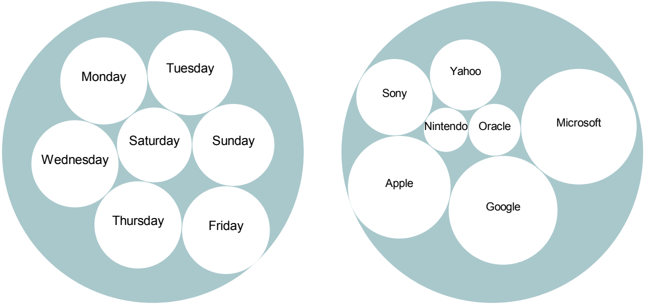
- Words can be grouped together into equivalence classes to help reduce data sparsity and better generalize the data.

- Words can be grouped together into equivalence classes to help reduce data sparsity and better generalize the data.





- Words can be grouped together into equivalence classes to help reduce data sparsity and better generalize the data.



- Hand-crafted equivalence classes are called **part-of-speech tags**, and automatically induced equivalence classes are usually called **word classes** or **word clusters**

Parts of Speech and Word Clusters

- Part-of-speech example:

Pierre	Vinken	,	61	years	old	,	will	join	the	board
NNP	NNP	,	CD	NNS	JJ	,	MD	VB	DT	NN

- Word cluster example:

Pierre	Vinken	,	61	years	old	,	will	join	the	board
344	0	283	94	348	274	283	367	360	71	390

Parts of Speech and Word Clusters

- Part-of-speech example:

Pierre	Vinken	,	61	years	old	,	will	join	the	board
NNP	NNP	,	CD	NNS	JJ	,	MD	VB	DT	NN

- Word cluster example:

Pierre	Vinken	,	61	years	old	,	will	join	the	board
344	0	283	94	348	274	283	367	360	71	390

Differences:

- Parts of speech have human-readable labels (eg. NN, VB), while word clusters usually just have numbers
- A word can have more than one part of speech (which depends on the context), while a word usually has just one word class

Supervised, Unsupervised, Semi-supervised Learning

- **Supervised learning** uses **manually-annotated data**

Supervised, Unsupervised, Semi-supervised Learning

- **Supervised learning** uses **manually-annotated data**

  - It's usually evaluated on **accuracy** (or related idea) of 'correct' label, given unannotated test input

Supervised, Unsupervised, Semi-supervised Learning

- **Supervised learning** uses **manually-annotated data**
  - It's usually evaluated on **accuracy** (or related idea) of 'correct' label, given unannotated test input
  - The data's[1] usually expensive and small

Supervised, Unsupervised, Semi-supervised Learning

- **Supervised learning** uses **manually-annotated data**

  - It's usually evaluated on **accuracy** (or related idea) of 'correct' label, given unannotated test input
  - The data's[1] usually expensive and small

- **Unsupervised learning** uses **unannotated data**

Supervised, Unsupervised, Semi-supervised Learning

- **Supervised learning** uses **manually-annotated data**
    - It's usually evaluated on **accuracy** (or related idea) of 'correct' label, given unannotated test input
    - The data's[1] usually expensive and small

- **Unsupervised learning** uses **unannotated data**
    - It's usually evaluated on the probability of the unannotated test input ($\propto$ **perplexity**), or a downstream task

Supervised, Unsupervised, Semi-supervised Learning

- **Supervised learning** uses **manually-annotated data**

  - It's usually evaluated on **accuracy** (or related idea) of 'correct' label, given unannotated test input
  - The data's[1] usually expensive and small

- **Unsupervised learning** uses **unannotated data**
  - It's usually evaluated on the probability of the unannotated test input ($\propto$ **perplexity**), or a downstream task
  - The data's usually big and noisy. Just like the world around us.

Supervised, Unsupervised, Semi-supervised Learning

- **Supervised learning** uses **manually-annotated data**

  - It's usually evaluated on **accuracy** (or related idea) of 'correct' label, given unannotated test input
  - The data's[1] usually expensive and small

- **Unsupervised learning** uses **unannotated data**

  - It's usually evaluated on the probability of the unannotated test input ($\propto$ **perplexity**), or a downstream task
  - The data's usually big and noisy. Just like the world around us.
- **Semi-supervised learning** uses **both** unannotated and annotated data
  - It's usually evaluated just like supervised learning tasks

[1] That's right: "data is".

Words *vs.* Word Classes

- Using word classes reduces the number of parameters in language models, which means **generalization**

Words *vs.* Word Classes

- Using word classes reduces the number of parameters in language models, which means **generalization**
- But, some sequences of words are **lexicalized**, meaning they are based on the specific words involved

Words *vs.* Word Classes

- Using word classes reduces the number of parameters in language models, which means **generalization**
- But, some sequences of words are **lexicalized**, meaning they are based on the specific words involved
- For example: "It's raining cats and ____"

Words *vs.* Word Classes

- Using word classes reduces the number of parameters in language models, which means **generalization**
- But, some sequences of words are **lexicalized**, meaning they are based on the specific words involved
- For example: "It's raining cats and ____"
- Using word-based language models, the next word will probably be '*dogs*'

Words *vs.* Word Classes

- Using word classes reduces the number of parameters in language models, which means **generalization**
- But, some sequences of words are **lexicalized**, meaning they are based on the specific words involved
- For example: "It's raining cats and ____"
- Using word-based language models, the next word will probably be '*dogs*'
- But class-based LMs only see something like "PRP VBZ VBG NNS CC ____"
- So they would predict something like '*shares*', if they were trained on the WSJ corpus

How Many Word Classes Should I Use?

The more word classes you use, the closer you get to a word-based model

How Many Word Classes Should I Use?

The more word classes you use, the closer you get to a word-based model

If you use a class-based LM by itself, more word classes is usually better, especially if you have a lot of training data

How Many Word Classes Should I Use?

The more word classes you use, the closer you get to a word-based model

If you use a class-based LM by itself, more word classes is usually better, especially if you have a lot of training data

However if you interpolate a class-based LM with a word-based LM, fewer word classes is usually better, because you get complementary information

How Can You Cluster Words?

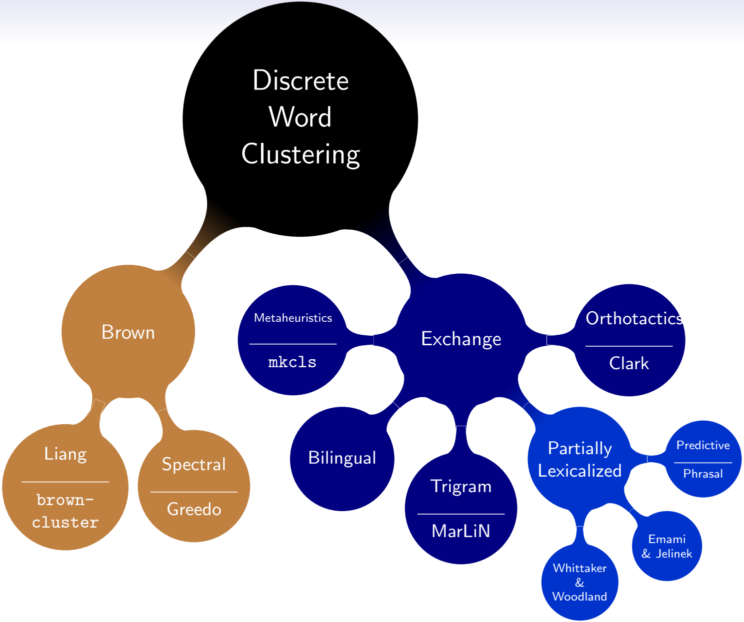
- If you represent words as vectors of real numbers, you can use general clustering algorithms like k-means clustering or agglomerative clustering

How Can You Cluster Words?

- If you represent words as vectors of real numbers, you can use general clustering algorithms like k-means clustering or agglomerative clustering
- You can also use discrete versions of these two algorithms, to cluster words directly from plaintext

How Can You Cluster Words?

- If you represent words as vectors of real numbers, you can use general clustering algorithms like k-means clustering or agglomerative clustering
- You can also use discrete versions of these two algorithms, to cluster words directly from plaintext
- Discrete agglomerative word clustering is usually called **Brown clustering**
- Discrete k-means word clustering is usually called **exchange algorithm clustering**



Discrete Word Clustering
Brown
Liang
brown-cluster
Spectral
Greedo
Exchange
Metaheuristics
mkcls
Orthotactics
Clark
Bilingual
Trigram
MarLiN
Partially Lexicalized
Predictive
Phrasal
Emami & Jelinek
Whittaker & Woodland

Brown Clustering (hierarchical clusters)

1. Assign the most frequent words to their own class

Brown Clustering (hierarchical clusters)

1. Assign the most frequent words to their own class
2. For the remaining words, assign the next most frequent word to the class giving the best likelihood of the training data

Brown Clustering (hierarchical clusters)

1. Assign the most frequent words to their own class
2. For the remaining words, assign the next most frequent word to the class giving the best likelihood of the training data
3. There is no step 3.

Brown Clustering (hierarchical clusters)

1. Assign the most frequent words to their own class
2. For the remaining words, assign the next most frequent word to the class giving the best likelihood of the training data
3. There is no step 3.
4. Optionally recursively merge the remaining classes, again based on likelihood

Brown Clustering (hierarchical clusters)

1. Assign the most frequent words to their own class
2. For the remaining words, assign the next most frequent word to the class giving the best likelihood of the training data
3. There is no step 3.
4. Optionally recursively merge the remaining classes, again based on likelihood

- Above is Percy-style Brown clustering, which can be more efficient than the original algorithm

Brown Clustering (hierarchical clusters)

1. Assign the most frequent words to their own class
2. For the remaining words, assign the next most frequent word to the class giving the best likelihood of the training data
3. There is no step 3.
4. Optionally recursively merge the remaining classes, again based on likelihood

- Above is Percy-style Brown clustering, which can be more efficient than the original algorithm
- The time complexity is $\mathcal{O}(|V| \times |C|^2)$
- $|V|$ is the size of the vocabulary
  $|C|$ is the number of word classes

Brown Clustering (hierarchical clusters)

1. Assign the most frequent words to their own class
2. For the remaining words, assign the next most frequent word to the class giving the best likelihood of the training data
3. There is no step 3.
4. Optionally recursively merge the remaining classes, again based on likelihood

- Above is Percy-style Brown clustering, which can be more efficient than the original algorithm
- The time complexity is $\mathcal{O}(|V| \times |C|^2)$
- $|V|$ is the size of the vocabulary
  $|C|$ is the number of word classes
- Thus it's fairly fast for small clusters (< 400), but slow for large clusters (> 800)

Exchange Algorithm (flat clusters)

1. Randomly assign each word to a class

Exchange Algorithm (flat clusters)

1. Randomly assign each word to a class
2. For each word, change its word class to the one giving the best likelihood of the training data

Exchange Algorithm (flat clusters)

1. Randomly assign each word to a class
2. For each word, change its word class to the one giving the best likelihood of the training data
3. Do the last step for a few times (maybe 10–20 iterations)

Exchange Algorithm (flat clusters)

1. Randomly assign each word to a class
2. For each word, change its word class to the one giving the best likelihood of the training data
3. Do the last step for a few times (maybe 10–20 iterations)

- Thus the basic time complexity is $\mathcal{O}(|V| \times |C| \times i)$
- $|V|$ is the size of the vocabulary
  $|C|$ is the number of word classes
  i is the number of iterations

Exchange Algorithm (flat clusters)

1. Randomly assign each word to a class
2. For each word, change its word class to the one giving the best likelihood of the training data
3. Do the last step for a few times (maybe 10–20 iterations)

- Thus the basic time complexity is $\mathcal{O}(|V| \times |C| \times i)$
- $|V|$ is the size of the vocabulary
  $|C|$ is the number of word classes
  i is the number of iterations
- There's a little more added complexity is how you calculate training-set likelihood

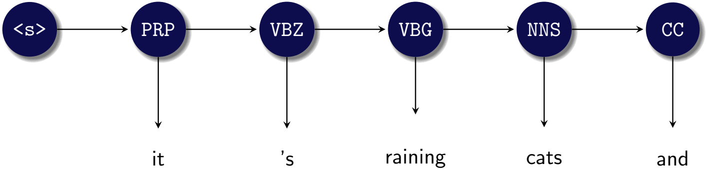
Class-based Language Models

- So how do we use word classes as a language model?

Class-based Language Models

- So how do we use word classes as a language model?
- The most common form (c_i is the word class of word w_i):

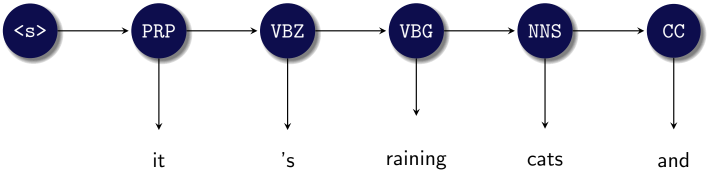
$$P(w_i|w_{i-1}) \triangleq P(w_i|c_i)\,P(c_i|c_{i-1})$$



Class-based Language Models

- So how do we use word classes as a language model?
- The most common form (c_i is the word class of word w_i):

$$P(w_i|w_{i-1}) \triangleq P(w_i|c_i)\,P(c_i|c_{i-1})$$



- Notice c_i, which is called a **bottleneck variable**
- The history is 'squeezed' through this point, in order to summarize and generalize the history

Predictive Exchange and Conditional Exchange

- The previous model is used in both Brown clustering and exchange algorithm clustering to determine the likelihood of the training set. We can use different models as well.

Predictive Exchange and Conditional Exchange

- The previous model is used in both Brown clustering and exchange algorithm clustering to determine the likelihood of the training set. We can use different models as well.
- The *predictive exchange algorithm* uses this model:

$$P(w_i|w_{i-1}) \triangleq P(w_i|c_i)\,P(c_i|w_{i-1})$$

- The *conditional exchange algorithm* uses this model:

$$P(w_i|w_{i-1}) \triangleq P(w_i|c_{i-1})$$

Uses of Word Clusters

Machine Translation

- Word alignment (Brown et al, 1993; Och & Ney, 2000)
- Factored/class-based translation models (Koehn & Hoang, 2007; *inter alia*)
- Reordering models (Cherry, 2013)
- Preordering (Stymne, 2012)
- Target-side inflection (Chahuneau et al, 2013)
- Syntax-augmented machine translation (Zollmann & Vogel, 2011)
- Sparse word features (Haddow et al, 2015)
- Operation sequence models (Durrani et al, 2014)

Uses of Word Clusters

Machine Translation

- Word alignment (Brown et al, 1993; Och & Ney, 2000)
- Factored/class-based translation models (Koehn & Hoang, 2007; *inter alia*)
- Reordering models (Cherry, 2013)
- Preordering (Stymne, 2012)
- Target-side inflection (Chahuneau et al, 2013)
- Syntax-augmented machine translation (Zollmann & Vogel, 2011)
- Sparse word features (Haddow et al, 2015)
- Operation sequence models (Durrani et al, 2014)

Other NLP Tasks

- Training Neural Net Lang. Models (Goodman, 2001; Mnih & Hinton, 2009; ...)
- Parsing (Koo et al, 2008; Candito & Seddah, 2010; Kong et al, 2014)
- Semantic Parsing (Zhao et al, 2009)
- Chunking (Turian et al, 2010)
- NER (Miller et al, 2004, *inter alia*)
- Tagging of Twitter Feeds (Owoputi et al, 2013; Nooralahzadeh et al, 2014)
- Structure Transfer (Täckström et al, 2012)
- Discourse Relation Discovery (Rutherford & Xue, 2014)

References I

Brown, P. E., Pietra, S. A. D., Pietra, V. J. D., and Mercer, R. L. (1993).
The mathematics of statistical machine translation: Parameter estimation.
Computational Linguistics, 19(2):263–311.

Candito, M. and Seddah, D. (2010).
Parsing word clusters.
In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 76–84, Los Angeles, CA, USA. Association for Computational Linguistics.

Chahuneau, V., Schlinger, E., Smith, N. A., and Dyer, C. (2013).
Translating into morphologically rich languages with synthetic phrases.
In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1677–1687, Seattle, WA, USA. Association for Computational Linguistics.

Cherry, C. (2013).
Improved reordering for phrase-based translation using sparse features.
In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 22–31, Atlanta, GA, USA. Association for Computational Linguistics.

Durrani, N., Koehn, P., Schmid, H., and Fraser, A. (2014).
Investigating the usefulness of generalized word representations in SMT.
In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 421–432, Dublin, Ireland. Association for Computational Linguistics.

Goodman, J. (2001).
Classes for fast maximum entropy training.
In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '01)*, volume 1, pages 561–564.

References II

Haddow, B., Huck, M., Birch, A., Bogoychev, N., and Koehn, P. (2015).
The Edinburgh/JHU phrase-based machine translation systems for WMT 2015.
In *Proceedings of the Tenth Workshop on Statistical Machine Translation (WMT)*, pages 126–133, Lisbon, Portugal. Association for Computational Linguistics.

Koehn, P. and Hoang, H. (2007).
Factored translation models.
In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 868–876, Prague, Czech Republic. Association for Computational Linguistics.

Kong, L., Schneider, N., Swayamdipta, S., Bhatia, A., Dyer, C., and Smith, N. A. (2014).
A dependency parser for tweets.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1001–1012, Doha, Qatar. Association for Computational Linguistics.

Koo, T., Carreras, X., and Collins, M. (2008).
Simple semi-supervised dependency parsing.
In *Proceedings of ACL-08: HLT*, pages 595–603, Columbus, OH, USA. Association for Computational Linguistics.

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

Mediani, M., Zhang, Y., Ha, T.-L., Niehues, J., Cho, E., Herrmann, T., Kärgel, R., and Waibel, A. (2012).
The KIT translation systems for IWSLT 2012.
In *Proceedings of the 9th International Workshop on Spoken Language Translation (IWSLT)*, pages 38–45, Hong Kong.

References III

Miller, S., Guinness, J., and Zamanian, A. (2004).
Name tagging with word clusters and discriminative training.
In Dumais, S., Marcu, D., and Roukos, S., editors, *HLT-NAACL 2004: Main Proceedings*, pages 337–342, Boston, MA, USA. Association for Computational Linguistics.

Mnih, A. and Hinton, G. (2009).
A scalable hierarchical distributed language model.
In Koller, D., Schuurmans, D., Bengio, Y., and Bottou, L., editors, *Advances in Neural Information Processing Systems 21 (NIPS)*, volume 21, pages 1081–1088.

Nooralahzadeh, F., Brun, C., and Roux, C. (2014).
Part of speech tagging for French social media data.
In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1764–1772, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Och, F. J. and Ney, H. (2000).
A comparison of alignment models for statistical machine translation.
In *Proceedings of the 18th International Conference on Computational Linguistics (Coling)*, pages 1086–1090, Saarbrücken, Germany.

Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., Schneider, N., and Smith, N. A. (2013).
Improved part-of-speech tagging for online conversational text with word clusters.
In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 380–390, Atlanta, GA, USA. Association for Computational Linguistics.

Ratinov, L. and Roth, D. (2009).
Design challenges and misconceptions in named entity recognition.
In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, CO, USA. Association for Computational Linguistics.

References IV

Ritter, A., Clark, S., Mausam, and Etzioni, O. (2011).
Named entity recognition in tweets: An experimental study.
In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534, Edinburgh, Scotland. Association for Computational Linguistics.

Rutherford, A. and Xue, N. (2014).
Discovering implicit discourse relations through Brown cluster pair representation and coreference patterns.
In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 645–654, Gothenburg, Sweden. Association for Computational Linguistics.

Stymne, S. (2012).
Clustered word classes for preordering in statistical machine translation.
In *Proceedings of the Joint Workshop on Unsupervised and Semi-Supervised Learning in NLP*, pages 28–34, Avignon, France. Association for Computational Linguistics.

Täckström, O., McDonald, R., and Uszkoreit, J. (2012).
Cross-lingual word clusters for direct transfer of linguistic structure.
In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 477–487, Montréal, Canada. Association for Computational Linguistics.

Turian, J., Ratinov, L.-A., and Bengio, Y. (2010).
Word representations: A simple and general method for semi-supervised learning.
In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, Uppsala, Sweden. Association for Computational Linguistics.

Wuebker, J., Peitz, S., Rietig, F., and Ney, H. (2013).
Improving statistical machine translation with word class models.
In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1377–1381, Seattle, WA, USA. Association for Computational Linguistics.

References V

Zhao, H., Chen, W., Kity, C., and Zhou, G. (2009).
Multilingual dependency learning: A huge feature engineering method to semantic dependency parsing.
In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 55–60, Boulder, CO, USA. Association for Computational Linguistics.

Zollmann, A. and Vogel, S. (2011).
A word-class approach to labeling PSCFG rules for machine translation.
In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 1–11, Portland, OR, USA. Association for Computational Linguistics.