Stony Brook University

Natural Language Processing using Word Connection Networks

A Dissertation presented

by

Yanqing Chen

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Computer Science

Stony Brook University

May 2015

Stony Brook University

The Graduate School

Yanqing Chen

We, the dissertation committee for the above candidate for the

Doctor of Philosophy degree, hereby recommend

acceptance of this dissertation

Steven Skiena - Dissertation Advisor
Distinguished Teaching Professor, Department of Computer Science

Niranjan Balasubramanian - Chairperson of Defense
Research Assistant Professor, Department of Computer Science

Andrew Schwartz
Assistant Professor, Department of Computer Science

Jiwon Yun
Assistant Professor, Department of Linguistics

This dissertation is accepted by the Graduate School

Charles Taber
Dean of the Graduate School

Abstract of the Dissertation

Natural Language Processing using Word Connection Networks

by

Yanqing Chen

Doctor of Philosophy

in

Computer Science

Stony Brook University

2015

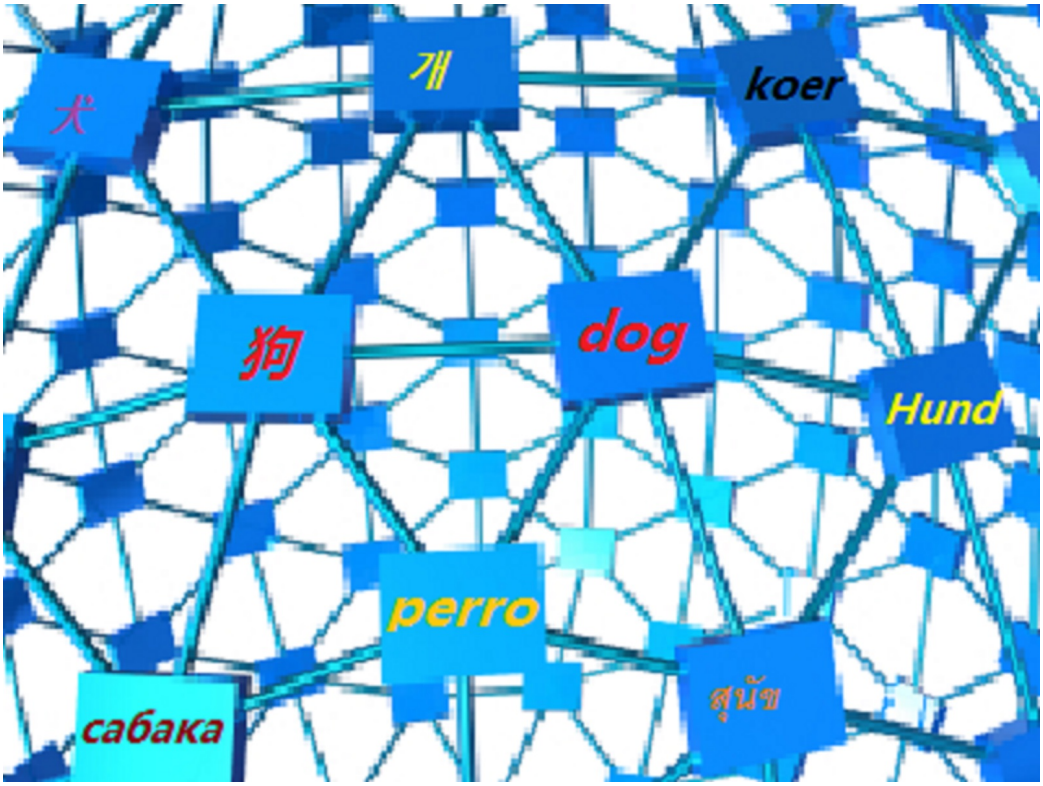
Word Connection Networks are graphs recording linguistic connections, including both semantic and syntactic connections, between single words. Specific Word Connection Networks of smaller sizes are frequently used in our daily communications – we search for counterparts of words in another language when doing translations and we group words by their sentiment when express feelings. Word Connection Networks are usually consistent with each other, which makes it an interesting and challenging idea to construct integrated language resources with both inter-language and intra-language connections to handle natural language processing tasks in a multilingual environment.

We propose to collect large-scale word-level linguistic resources from the web that reflect qualitatively different types of connections between words across major languages and integrate them into Word Connection Networks. Our data sources include translations from online machine translation systems, transliterations of entities across major languages, semantic relationships between words from human annotations, distributed word representations which captured both semantic and syntactic features out of raw text and quantified sentiment polarities from sentiment analysis researches / applications. These resources cover different aspects of language features and

contribute to the completeness of Word Connection Networks; thus we have strong and versatile knowledge bases to handle generalized natural language processing tasks. Additionally, we do research on numbers, frequently appearing but usually being ignored in language tasks, to explore word-level features inside their existence.

The core contributions of this thesis are deeper knowledge mining in Word Connection Networks and extensions to generate valuable resources for various natural language processing tasks. Implementation of Word Connection Networks allows quantifying expressive power of connections from difference sources in a specific task. We make each single connection in Word Connection Networks traceable and implement a propagation method for information transitivity inside the graph, which allows us to discover a high-confidence model of semantic or syntactic connections that does not currently exist. We prove that inter-language connections preserve good features on word level from more detailed intra-language connections. We successfully finished several natural language processing tasks using connections in Word Connection Networks and we have generated new resources, including high frequency sentiment lexicons for 136 major languages and transliterations of 69 languages, by applying graph algorithms on Word Connection Networks.

This dissetation I dedicate
to
my beloved family members



Natural Language Processing using
Word Connection Networks

Table of Contents

List of Figures **ix**

List of Tables **xvi**

Acknowledgements **xxi**

1 Introduction to Word Connection Networks **1**
1.1 Functions of Word Connection Networks 2
1.2 Constructing Word Connection Networks 4

2 The Expressive Power of Word Representations **8**
2.1 Our work 9
2.2 Related work 11
2.3 Experimental setup 13
2.3.1 Evaluation tasks 13
2.3.2 Embedding datasets 14
2.3.3 Classification 15
2.4 Evaluation Results 15
2.4.1 Term Classification 15
2.4.2 Pair Classification 17
2.5 Information reduction 20
2.5.1 Bitwise truncation 21
2.5.2 Principle component analysis (PCA) 23
2.6 Application of word representations 26

3 Constructing Multilingual Sentiment Lexicons **27**
3.1 Our work 28
3.2 Related work 29

3.3 Resource Statistics . . . 30
3.3.1 Statistics of Word Connection Networks . . . 30
3.4 Graph Propagation . . . 32
3.4.1 Why not Label Propagation? . . . 34
3.4.2 Source English Sentiment Lexicons . . . 35
3.4.3 Parameterization of Edge Classes . . . 35
3.5 Lexicon Evaluation . . . 36
3.5.1 Testing Score . . . 36
3.5.2 Per-language Analysis . . . 38
3.6 Extrinsic Evaluation: Consistency of Wikipedia Sentiment . . . 40
3.6.1 Normalizing across languages . . . 41
3.6.2 Consistency between language pairs . . . 43
3.7 Application of multilingual sentiment lexicons . . . 46

4 True Friends and False Friends: Digging Deeper into Transliterations 48
4.1 Our work . . . 49
4.2 Related work . . . 51
4.3 Data collection and pre-processing . . . 52
4.4 Training transliteration model . . . 54
4.5 Glances at character matching . . . 56
4.6 Experimental Results . . . 58
4.6.1 Baseline Model . . . 58
4.6.2 Test Results . . . 60
4.7 True and false friends detection . . . 61
4.7.1 Evaluation against Human Annotation . . . 64
4.7.2 Cross-Language Scan . . . 64
4.7.3 Cross-Language Validation . . . 66
4.8 Transliteration and Word Connection Networks . . . 67

5 Numbers: Standing out of the Multilingual World 70
5.1 Our work . . . 71
5.2 Related work . . . 71
5.3 Data collections and methodology . . . 72
5.4 Time series comparison of different representation of numbers 74
5.5 Preference of last digits . . . 77
5.6 Quantities with units . . . 80
5.6.1 Discussion of weight . . . 80

5.6.2 Discussion of lengths 82
5.6.3 Discussion of currency 84
5.6.4 Discussion of time 85
5.6.5 Quantities without defined unit systems 87
5.6.6 Benford's law 90

6 Comparing Historical Figures using Wikipedia 94
6.1 Related work 97
6.2 Data collection 98
6.3 Model description 99
6.3.1 TF-IDF model 99
6.3.2 Distributed word embedding model 99
6.3.3 LDA model 101
6.3.4 Deepwalk embedding model 102
6.4 Wikipedia categories processing 102
6.4.1 Constructing reference standard 103
6.5 Which feature vector is better? 106
6.6 Descriptive power of Wikipedia categories 108
6.6.1 Collecting human annotations 109
6.6.2 Definition of close neighbors 111
6.6.3 Ranking a category 112
6.7 Evaluating category ranking 113
6.7.1 Influence of feature vectors 114
6.7.2 Influence of distance measurement 115
6.7.3 Influence of defining close neighbors 115
6.7.4 Influence of importance measurement 116
6.8 Categories with bad performances 117
6.9 Conclusion 118

List of Figures

1.1 Illustration of our Word Connection Networks. Edge representation keeps track of links between words and preserves source identity. For each edge between corresponding word pair, we encode an integer recording the existence of possible semantic links. 6

2.1 Numerical (top) and color (bottom) representation of sample words. Word representation can group words with similar syntactic and semantic behaviors together but it is hard for human to learn specific knowledge inside word representations from numerical format. Color format converts floating numbers to red-blue scale to better capture distances between words. For instance, cities are close with each other, slightly faraway from ordinary noun "tree" and distant from verb "run" in high dimensional space. 9

2.2 2-D visualization sample of word representations. Compared with raw formats in Figure 2.1, correct processing including projection and dimension reduction provides much better visualization for human to learn specific knowledge inside word representations. 11

2.3 Results of term-based tasks. To illustrate that strong performance is still possible on such tasks, we report results by classifier type separately. Unshaded areas show average results from the tasks across classifiers using the geometric mean. Shaded areas represent improvements using kernel SVM. 16

2.4 Comparison between treating Region spelling task as term-based test and as pair-based test. The result shows that difference between embedding encoded more information than absolute positions in embedding space. 18

2.5 Results of pair-based tests on 2-class classification tasks. Unshaded areas show average results from the tasks across classifiers using the geometric mean. Shaded areas represent improvements using kernel SVM. 19
2.6 Performances on the 3-class version of Sentiment task. Unshaded areas show average results from the tasks across classifiers using the geometric mean. Shaded areas represent improvements using kernel SVM. 20
2.7 Performances on the 3-class version of Synonyms and antonyms task. Unshaded areas show average results from the tasks across classifiers using the geometric mean. Shaded areas represent improvements using kernel SVM. 21
2.8 Results of reducing the precision of the embedding, averaged by the geometric mean of each dataset. Notice that after removing 31 bits, each dimension of the embedding is a binary feature. 22
2.9 Results of reducing the precision of the embedding, averaged by the geometric mean of tasks. 23
2.10 Results of reducing the dimensions of the embeddings through PCA, averaged by the geometric mean of each dataset. 24
2.11 Results of reducing the dimensions of the embeddings through PCA, averaged by the geometric mean of each task. 25
2.12 Comparison of performance changes between linear classifiers and nonlinear classifiers during PCA. 25

3.1 Count of different edges represented in log format of 10 biggest Wikipedia languages. English acts as the hub of all languages thus it contains many google translation links. Transliteration links provides word pairs with spelling and sound similarity. Wiktionary links provides semantic connections that are not discovered by Google translations. 31
3.2 Illustration of label propagation and belief propagation. Initial Red / Green nodes represent positive / negative seed words. Blank nodes will be affected via links during propagation, gaining sentiment of close neighbors. Label propagation resolves conflictions after each round belief propagation record only the best path in parallel and resolve conflictions once at the end. 33

3.3 Accumulated distribution of sentiment word count. About 20 languages have less than 100 propagated sentiment words. Totally 60 languages have less than 1,000 sentiment words. . . 40
3.4 Correlation between Wiktionary entries and sentiment word count. Languages with less than 5,000 Wiktionary entries are usually those we cannot find good propagations of sentiment lexicons, indicating that better language resources are needed for these languages. 41
3.5 Sentiment score distribution of Wikipedia pages between English and (left) Spanish / (right) French. We calculate sentiment scores using our propagated sentiment lexicons. The green line estimated correlations and we show that strong correlations exist in scores of the same person but different language version. 42
3.6 Distribution of Wikipedia pages in 10 biggest Wikipedia languages. Top subfigure shows absolute sentiment value. Bottom subfigure mitigated differences in lexicon polarity composition by normalization, enabling the results to be compared directly across languages. 43
3.7 Heatmap showing sentiment correlation between 30 Wikipedia languages on 2,000 most famous historical figures according to *Who's bigger?* [88]. First 10 languages are 10 biggest languages discussed in previous sections. 44

4.1 Constructed best and detected best for word "obama" where capitalization is disabled. Constructed best is generated via cost matrices without any prior knowledge of vocabulary. Detected best is the best match in 100,000 most frequent words in Wikipedia. Last column shows the rank of gold standard reference if it appears in these 100,000 high frequency words. . 50
4.2 Illustration of the training procedure. Each round we compute minimum-cost-matching and record matched string pieces for all training examples and update costs matrix through a Bayesian model. 55
4.3 Probability/cost matrix for single character pairs between English and a) French, b) Spanish, c) Russian and d) Hebrew. The bright diagonal shows that we discover common equivalence for most Latin characters. 57

4.4 Best matches of single English character in some non-Latin languages, grouped by language families to show consistency between close languages. Languages do not always follow one on one character matching rule (e.g. Korean) but there still exist high correlation between listed character pairs. 59
4.5 Distribution of sound edit-distance from English to one other language. Figure 4.5b shows the distribution of distance measured by our transliteration system. We discovered gaps near distance = 2.0 since the initial cost of substitute one letter with an arbitrary letter is 2.0. Figure 4.5a demonstrates the percentage that close pairs judged by our transliteration system match with Google translations. 62
4.6 Illustration of word embedding test. In case no direct evidences of semantic similarity between "Cat" and "Gato" are found, we check number of translations that links nearest neighbors of "Cat" and "Gato" . Since ("Dog", "Monkey", "Duck") matches perfectly with ("Perro", "Mono", "Pato"), we can judge that ("Cat", "Gato") has close semantic meanings. ("Car" , "Gato") will definitely fail this test. 63
4.7 Detailed examples of true and false friends between English and Russian. Right part lists words with low sound edit-distances. These words are easily confused according to their pronunciations. Left part shows words with slightly high edit distances, affected by small sound changes like inflections. Top part lists words that are semantically identical while lower parts shows words having different meanings. We demonstrate that there are high correlations between judgements from our transliteration model and sentiment similarity measurement and which part the words fall in. 66
4.8 (top) Fraction of gold standard translations within very close edit distance pairs ($d < 2$) versus the next closest 10,000 pairs. (bottom). Same fractions after retaining only the 50% of pairs which are closest by embedding distance. For 68 of 69 languages, the lexically closer pairs are more likely to be translations (top). Further, eliminating pairs failing the embedding test shifts all languages to the upper right, showing that the embedding test accurately captures semantic similarity (bottom). 68

5.1 Time-series showing the trend of different representations of numbers. The category "integers numbers" contain only integers formed by digits, "floating numbers" contain real numbers formed by digits and floating point, while "text numbers" represent number in the form of English words, i.e. "nine" . . 75
5.2 Time series of text-specified numbers, with conditions of disabling some of them. The top line is the original time series, the middle line shows the situation if we discard the word "one". The bottom line throw away all numerical words from "one" through "nine". 76
5.3 Time series of mean and median values of (top) 3-digit-numbers and (bottom) 4-digit-numbers. The distribution of 3-digit-numbers basically obey Benford's law, supporting the fact that people does not have specific usage of 3-digit-numbers in real-life. However, the usage of 4-digit-numbers usually correlate with years, especially current years. 78
5.4 Percentage of numbers ending with 0 or 5 in different categories. Even of small numbers (i.e. less than 3 digits), there are about 30% of them ending with 0 or 5, showing a significant difference on random baseline which is 20%. 3-digit-numbers and 4-digit-numbers get closer to average due to the need of precision (i.e. year). 79
5.5 Accumulated distribution of weights in (top) SI mass system, unit gram and (bottom) English mass system, unit pound. The trend of accumulation seems similar but the gap position indicates that there exist huge differences in how people use these two systems in daily life. 81
5.6 Accumulated distribution of lengths in (top) SI length system, unit meter and (bottom) English length system, unit foot. We see similar distributions and changing points as the weight system. 83
5.7 Compare distributions of object lengths between different length systems. Dots show lengths distribution using SI units within the range of 1 meter to 100 kilometers. Lines show the lengths distribution within the same range but use corresponding English units. 84

5.8 Accumulated distribution of US currency, unit dollar. The shape is totally different from what we see in weight systems and length systems as human world of currency changed a lot in past 100 years. 85
5.9 Accumulated distribution of time, unit minute. Longer time periods are mentioned more during the development of science and technologies. 86
5.10 Distribution of numbers appearing before the word "people". There are some outliners talking about meaningless fraction number of people, however majority of values ranges from $10^0 = 1$ as expected. 88
5.11 Expectation and observation of numbers starting with different leading digits that connect to unit "meter", grouped by five 40-year-periods. We find only minor gaps between expectations and observations except the earlist period of 1800-1840. 90
5.12 A stacked area graph showing the verification of Benford's law. Each strip represents the percentage of numbers starting with a certain digit, in the order of 1, 2, 3, ... , 9 from bottom to top. 91
5.13 Deviation from expectation over time indicating when the usage of numbers suddenly changes. Peaks usually reflects the process of replacing old arithmetic habit or increasing the digits used for counting due to development of businesses. 92
5.14 A stacked area graph showing the distribution of all numbers regardless of the units. Layers from bottom to top represent 1, 2, 3, ... , 9. We see clearly from the graph that smoothed distributions of numbers match pretty well with Benford's law, indicsating that most of the numbers are used for naturally developing businesses. 93

6.1 Sample analogous historical figures of Isaac Newton and corresponding explanations of similarity. Analogies are highly subjective thus it is impossible to find perfectly fair and objective gold standards. 95
6.2 Sample entities and words example in projected embedding space, distance between pairs shows their relevance. Entities will be attracted by the most descriptive words and locate close to the "centroid". 100

6.3 Examples of entities and top six related topics in LDA method. We demonstrate each topic by showing representative words. The probability of topic distribution differ a lot for party leaders, musician and physicists. 101
6.4 Distribution of Wikipedia category numbers on pages of people. It is clear that we usually have more detailed information on famous historical figures and the category comparison could be more precise. More famous people usually occupy more Wikipedia categories. Overall average categories between people lies between 8 and 9. 104
6.5 Question samples to collect human wisdom. For each question we collect 20 answers and the distribution of these answers indicate overall importance of each category. 110
6.6 Different definition of close neighbors and corresponding best performance. We experimented strategies of making a fixed number of close neighbors for each point as well as creating a comparable number of close neighbors overall using a distance limitation. Fixed number strategy outperforms the distance definition. 116

List of Tables

1.1 Close pairs suggested by phylogenic language trees. Languages on the right are the closest language in our 136 candidates to the corresponding language on the left. Notice that such relationships might not be symmetric and some language may not have a clear closest neighbor (e.g. Chinese). 5

2.1 Example data input for each task. Top 3 tasks are term-wise comparisons accepting embedding of only one word as input. Bottom 2 tasks are pair-wise comparisons and we feed two word representations at a time. 14

2.2 Most positive and negative examples using logistic regression on Sentiment task and Regional spelling task. *Prob* column measures how well the example fit into one category. Word like *resilient* could have positive and negative connotations in text and we find it close to the region were the words are more neutral than being polarized. 17

3.1 Statistics of Word Connection Networks. *Isolated vertices* are vertices without any semantic links in Word Connection Networks. *Synonyms edges* and *Antonyms edges* only consider those in English from *SentiWordNet*. 30

3.2 Network statistics of 10 biggest Wikipedia languages. We calculated fraction of isolated vertices as well as average vertex degrees by edge source to give a brief idea of the structure of Word Connection Networks. English works as a hub in collecting translations from and to other languages so it has a great number of average degrees. 31

3.3 Edge parameter weights and corresponding performances. *Best* column demonstrates the optimized parameters via grid search. Following columns each shows values and performances of weakening a certain type of edges. Performances are measured via the agreement of our test dataset of published non-English sentiment lexicons. 36
3.4 Belief propagation vs label propagation, using Liu's lexicons as seed words. **Accuracy** represents the ratio of identical polarity between our analysis and the published lexicons. **F1** combines precision and recall. **Coverage** reflects what faction of our lexicons overlap with published lexicons. 37
3.5 Belief propagation vs label propagation, starting from SentiWordNet. Measure metrics have the same definition as Table 3.4. 38
3.6 Statistics of propagated sentiment lexicons in major languages. We tag 10 languages having most/least sentiment words with blue/green color and 10 languages having highest/lowest ratio of positive words with orange/purple color. **Count** shows number of propagated sentiment lexicons and **Ratio** demonstrates the ratio of positive lexicons. 39
3.7 Z-score distribution examples of typical "good guys", "bad guys", "neutral guys" and "biased historical figures". We label 10 languages with their language code and other using tick marks on the x-axis. It is obvious that our Z-score measurement keep good consistency across languages in categorizing good and bad people. 45

4.1 Languages with the largest and smallest set of possible entities transliterations, i.e. reflecting the availability of training data. 54
4.2 10 cleanest and dirtiest languages, defined according to the ratio of flawed examples (i.e. those cannot find correlations of transliterations) in the training set. 56

4.3 Comparison of performances on Wikipedia and third party (TP) datasets. Top-k measures the percentage of correct transliterations in the top k candidates. Levenshtein 1 measures the percentage of the highest ranked transliteration that is no more than 1 substitution away from the reference transliteration, given that we consider insertion / deletion to be a special kind of substitution. 61
4.4 Accuracy and F1 score for our true and false friends distinguisher using Google translation and Word representation. . . 64
4.5 Statistics of True and False Friends between English and all 69 languages. TP denotes Google translation pairs which are not close in our word embedding. EP denotes close embedding pairs not recognized as translations by Google, while B denotes words pairs passing both semantic tests, N denotes false friends: word pairs which pass neither semantic test. Green languages are those with the highest ratio of B / TP, showing a significant correlation between our embedding and Google translation. At least 50% of Google translation pairs survives our embedding test for all 33 **Bolded** languages. By contrast, the Red languages are those where the embedding test performed poorly. 65

5.1 Statistics of most popular words after numbers in different time periods in past 200 years. 89

6.1 Examples showing three cases of comparing different similarity levels. (X, Y) is always closer than (X, Z) based on counts of common Wikipedia categories. The accuracy will be judged by how well we recover such comparisons from our vectors. . . 105
6.2 Accuracy performance of candidate models with different parameters. 106
6.3 Examples of 10 closest neighbors we find using our vector based models and comparison to our reference standard. *C* column represents count of common Wikipedia categories between pairs of people and *HE* column shows human evaluation after reading their bibliography, if it is a GOOD, OK or BAD match according to general knowledge. 109

6.4 10 most confusing category pairs in our questions. *Co-Prob* of two categories A and B is defined as the geometric mean of $P(A|B)$ and $P(B|A)$. Since candidate choices are manually picked from existing Wikipedia categories, such co-occurrence reflect some level of general human background knowledge preference. 111
6.5 Best performance of each feature vectors. Deepwalk outperforms the others and achieve 74.64% overall accuracy and 46.47% agreement with human top votes. *Corresponding distance* gives the threshold of distance to guarantee a certain average of close neighbors for overall. 114
6.6 Best performance of each feature vectors and distance measurement combinations. *Best Overall* shows the best performance achievable and the last 5 columns demonstrate per-rank agreement. There are no big gaps between L1 normalization and L2 normalization for all four types of feature vectors. Also, JS divergence does not yield better performances. 115
6.7 Best performance of each feature vectors and importance level measurement. Surprise level measurement outperforms Tightness, indicating that the size of category should be carefully examined during the procedure. 117
6.8 Categories disagree most with votes. *Count* shows number of times human vote for this category but not higher-ranked ones. *Probability* shows the chance of this category being answered whenever it appears. 118

Acknowledgements

First off, I wish to express my sincere thanks to my advisor, Steven Skiena. The thesis would never have been finished, without his help, guidance, and encouragement.

I take this opportunity to express gratitude to all the faculty members at Stony Brook I have interacted with, including my Research Proficiency Exam (RPE) committee members, thesis proposal committee members and thesis defense committee members. Indeed I received very useful suggestions from all of them during my researches.

I am extremely grateful to my lab mates Charles Ward, Rami Al-Rfou, Bryan Perrozi and Vivek Kulkarni. I am indebted to them for sharing great thoughts during our discussions on related topics.

I would also like to show my gratitude to all other colleagues in the Data Sciences Lab I have worked with, including Rukhsana Yeasmin, Yingtao Tian, Haochen Chen, Yafei Duan, Zichao Dai, Vincent Tsuei, Dhruv Matani, Samudra Banerjee and Ajeesh Elikkottil. Without their collaboration and support I could not finish my research work so easily.

The work in this thesis was supported with funding from The Research Foundation of SUNY and National Science Foundation.

Chapter 1

Introduction to Word Connection Networks

Word level features are useful in many natural language processing tasks. For instance, WordNet [35] provides per-sense annotations and synonyms / antonyms relationships in English and proved to be useful in tasks of word sense disambiguation, information retrieval, automatic text classification, automatic text summarization and machine translation.

However, currently we have no existing word level connections resources in multilingual world. It is interesting to construct integrated language resources with large-scale inter-language and intra-language connections to handle natural language processing tasks in multilingual environment.

We propose to make Word Connection Networks, graphs that record linguistic connections, including both semantic and syntactic connections as edges, between single words as vertices. Unlike other language resources that emphasize grammar level or sentence level, Word Connection Networks focus on word level features which are consistent and language-independent in multilingual environment. Key research of Word Connection Networks is to utilize multiple semantic and syntactic relationships between words and store them using various connections as edges in the graph. For instance, we will have connections of translations, synonyms and antonyms to generalized sentiment analysis in multilingual world since these connections preserve sentiment features of words. We also have connections of script matching and transliterations that can be easily extended to different languages and help named entity recognitions and co-reference resolutions.

Word level features are usually consistent across languages. With graph-

based implementation, we can quantify expressive power of connections in a specific task and resolve potential conflicts, for instance, adjust confidence level according to local agreement in the network. Additionally, we can mine deeper on existing knowledge to discover new semantic or syntactic connections based on transitivity of corresponding information. Such newly created high-confidence connections could be valuable towards natural language processing tasks.

The challenging parts of this work include 1) huge size of lexicons in multiple languages, 2) difficulty in generalizing resources and 3) lack of evaluation metrics in multilingual world. Constructing such a huge network is non-trivial because of its scale since we have to collect valuable human annotations and results from online machine learning for about 100,000 most frequent words in each of 136 languages to guarantee both coverage and accuracy. We have to integrate different format of resources which include but are not limited to translations, sentiment similarities, semantic resemblances, transliterations. It is also reasonable to store single-word-features on vertices, such as part of speech (POS) tags and distributed word representations. Last but not least, there are few publicly available language resources that could be compared against, especially for smaller language or dialects. We have to do some tricks that utilize available resources to give a reasonable evaluation in multilingual environment.

1.1 Functions of Word Connection Networks

Our motivation suggests that Word Connection Networks should have the following functions:

- **Keeping track of information from various sources** – Word Connection Networks need to keep track of as many valuable connections between words as possible. Useful pair-wise relationships between words include sound similarities, semantic resemblances (e.g. synonyms and antonyms), orthographical similarities and translations. All these connections can fit into specific natural language processing tasks. Plus, each type of connection allows basic transitions that can be used to reach unknown words. We also keep some single-word features like POS tags and word representations for potential uses in different language tasks.

- **Conflicts resolving and relationship passing** – It is important to coordinate with existing language resources in Word Connection Networks . Connections between words are usually with different expressive power. Whether connections of certain type should be weakened or enhanced are based on tasks. For instance, "fresh" usually acts as a positive sentiment lexicon (e.g. "fresh" fruits are good) but sometimes it points to a negative connotation (e.g. a "fresh" man is not experienced). Resolving such kind of conflict means better utilization of words and can greatly improve performances.

- **Extensibility** – We cannot manually fill in all information in Word Connection Networks. However, we can simply learn new knowledge based on current observations since information store in connections are usually transitive. Graph propagation method are designed for such demands. According to local graph structures, we can easily apply certain kind of propagation and create high-confidence connections between target vertices.

From statistics of WordNet [35], on average a word would have less than 3 senses, thus corresponding number of semantic / syntactic connections are rather small compared with size of vocabulary in the dictionary. We observe that sparse connections in Word Connection Networks show great advantages in storing and optimizing local graph algorithms as well as recording more valuable information with simple changes of encoding in data structure. Word Connection Networks cover more aspects as we gather more reliable resources and add them into the network.

On the other hand, when applying algorithms like graph propagation, Word Connection Networks can pass features from specific seed words to undocumented part of the graph via different connections. We successfully apply graph propagation algorithms to create sentiment lexicons for 136 major languages, proved that our Word Connection Networks can generate useful information with an acceptable error rate and thus greatly increase the potential of discovering various new resources across different languages in the world.

Last but not least, Word Connection Networks offer excellent opportunities to improve current language resources. Word representations for multi-sense words are dominated by their major usages. However, for some words with evenly important POS tags, for instance, "round" and "run" (noun, verb), the final representations will be averaged as act as an outliner to both

groups of nouns and groups of verbs. Research shows that word representations trained with human inferences provides better features of multi-sense words [46], it will be useful to discard low-quality word representations according to human annotations in Word Connection Networks.

1.2 Constructing Word Connection Networks

This part describes how we leverage off a variety of NLP resources to construct the framework of Word Connection Networks. Vertices in the network are designed to represent vocabularies in major languages. However, we definitely cannot represent all of them. It is required to pick a reasonable number of representatives, balancing coverage of words and connections as well as space of storage. As a reference, the Polyglot project [6] identified 100,000 most frequent words in each language's Wikipedia, showing a high coverage on web texts. Drawing a candidate lexicon from Wikipedia has some downsides, for instance, limited observations of informal words, but such lexicons are representative and convenient for a large number of languages. In particular, we collect a total of 7,741,544 high-frequency vocabulary words from 136 languages to serve as vertices in our graph.

Edges should record possible semantic / syntactic connections between words in different languages. Word Connection Networks adopt following resources:

- **Wiktionary** – This growing resource has entries for 171 languages, edited by people with sufficient background knowledge of specific language. Wiktionary provides translations covering 382,754 vertices in our graph.

- **Machine translation** – We script the Google translation API to get even more translations connections. We make English as a hub, in particular we ask for translations of each word in our English vocabulary to or from 57 languages with available translators and also we ask for translations of each word in 57 non-English dictionary to or from English. Additionally, we use the phylogeny of languages to identify 35 closely related pairs of languages [103], such as Turkish and Azerbaijani, to enrich our internal links. We believe close semantic connections between close language pairs coordinate better. Details

of close language pairs can be found in Table 1.1. In total, machine translation establishes connections between 3.5 million vertex pairs.

Language pairs		Language pairs		Language pairs	
Afrikaans	Dutch	Albanian	Armenian	Arabic	Maltese
Armenian	Lithuanian	Azerbaijani	Turkish	Belarusian	Russian
Bengali	Marathi	Bosnian	Croatian	Bulgarian	Macedonian
Catalan	Spanish	Cebuano	Filipino	Croatian	Serbian
Czech	Slovak	Danish	Swedish	Dutch	Afrikaans
Estonian	Finnish	Filipino	Cebuano	Finnish	Estonian
French	Catalan	Galician	Portuguese	German	Yiddish
Greek	Armenian	Hebrew	Arabic	Hindi	Urdu
Hungarian	Finnish	Icelandic	Swedish	Indonesian	Malay
Irish	Welsh	Italian	Spanish	Japanese	Korean
Kannada	Tamil	Khmer	Vietnamese	Korean	Japanese
Lao	Thai	Latin	Spanish	Latvian	Lithuanian
Lithuanian	Latvian	Macedonian	Bulgarian	Malay	Indonesian
Maltese	Arabic	Marathi	Bengali	Norwegian	Danish
Persian	Marathi	Polish	Czech	Portuguese	Galician
Romanian	Italian	Russian	Ukrainian	Spanish	Portuguese
Serbian	Bosnian	Slovak	Czech	Slovenian	Bosnian
Swedish	Danish	Tamil	Kannada	Thai	Lao
Turkish	Azerbaijani	Ukrainian	Belarusian	Urdu	Hindi
Vietnamese	Khmer	Welsh	Irish	Yiddish	German

Table 1.1: Close pairs suggested by phylogenic language trees. Languages on the right are the closest language in our 136 candidates to the corresponding language on the left. Notice that such relationships might not be symmetric and some language may not have a clear closest neighbor (e.g. Chinese).

- **Script matching** – Natural flow brings words across languages with little morphological change. Closely related language pairs (i.e. Russian and Ukrainian) share many characters/words in common. Though not always true, words with exact same spelling usually have similar meanings so this can improve the coverage of semantic links. Transliteration provides more than 36 million links in Word Connection Networks.
- **WordNet** – We gather synonyms and antonyms of English words from

WordNet [35], which prove particularly useful in propagating sentiment across languages. In total we collect over 100,000 pairs of synonyms and antonyms.

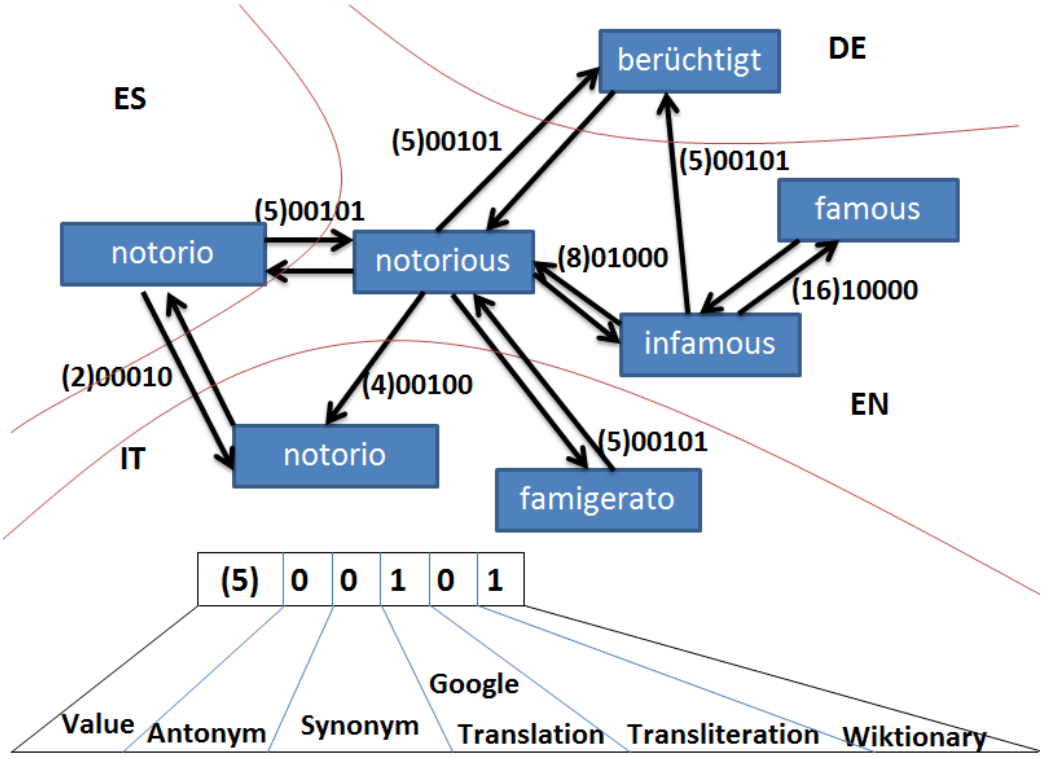


Figure 1.1: Illustration of our Word Connection Networks. Edge representation keeps track of links between words and preserves source identity. For each edge between corresponding word pair, we encode an integer recording the existence of possible semantic links.

Figure 1.1 illustrates how we encode semantic relationships between two words into integer value of corresponding edge. We store relationships from a specific resource as a certain type of links. Though links usually agree in both directions, we do not expect a bidirectional graph at the end. Multi-sense words, particularly, link to different targets as the sense changes. To avoid losing information, we use unidirectional links in our word translation network.

In next few chapters we will describe various multilingual features in Word Connection Networks as we expect to integrate as many useful resources as possible. Chapter 2 is about word representations we learned from Wikipedia with its power to summarize and predict semantic / syntactic relationships of word pairs. Chapter 3 describes our graph propagation method and how we create sentiment labels for vertices in the graph. Chapter 4 talks about an advanced text-based transliteration system with higher accuracy and coverage on detecting borrowed words. Chapter 5 studies historical trend of numbers, trying to figure out importance of numbers since numbers have a stable expressive power across all possible languages. Chapter 6 describe an application of finding historical analogous which uses knowledge integrated in Word Connection Networks.

Chapter 2

The Expressive Power of Word Representations

Distributed word representations (a.k.a. word embedding) are high dimensional numerical vector representation of words. Such representations are trained via large amount of corpus [94, 6] without human intervention or language dependent processing. Word representations are expected to capture semantic and syntactic features of words from large amount of context. Training word representations are usually unsupervised and features captured by embedding are task independent, which make them ideal for language modeling and setting up inner language semantic or syntactic connections. Word representations are fixed resources once training is complete and can be directly applied to various natural language processing tasks.

However, embedding is hard to interpret and understand since points in high dimensional spaces carry a lot of information that is hard to quantify as shown in Figure 2.1. The efforts of visualizing the word embedding [96] show one possibility. Figure 2.2 [1] shows that without appropriate processing (e.g. 2-D projection), it is difficult to reveal hidden connections in word representations. Additionally, publicly available embedding generated by multiple research groups use different data and training procedures and there is not yet an understanding about the best way to learn these representations. We did a detailed study to extract, analyze and explain information inside word representation.

[1] Perozzi, B, Al-Rfou', R, Kulkarni, V and Skiena, S. Inducing Language Networks from Continuous Space Word Representations. Complex Networks V, volume 549, page 261–273, 2014.

Word	Numerical Representation
	n dimensions
detroit	$\overbrace{0.20, 1.76, 0.25, ..., 0.61, 0.57, -1.20}$
chicago	0.11, 0.93, 0.87, ... , -0.33, 0.07, -0.27
seattle	0.46, 0.45, 0.51, ... , 0.46, 0.39, -0.55
tree	1.54, 1.76, -1.27, ... , -0.10, 0.00, 0.68
run	-0.55, 0.28, -0.48, ... , -0.54, -0.87, 0.65

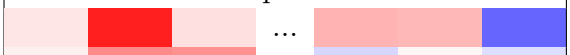
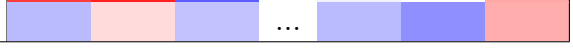
Word	Color Representation
detroit	
chicago	
seattle	
tree	
run	

Figure 2.1: Numerical (top) and color (bottom) representation of sample words. Word representation can group words with similar syntactic and semantic behaviors together but it is hard for human to learn specific knowledge inside word representations from numerical format. Color format converts floating numbers to red-blue scale to better capture distances between words. For instance, cities are close with each other, slightly faraway from ordinary noun "tree" and distant from verb "run" in high dimensional space.

2.1 Our work

Our published work [23] [2] investigate four public released word embedding: (1) HLBL, (2) SENNA, (3) Turian's and (4) Huang's. We use context-free classification tasks rather than sequence labeling tasks (such as part of speech tagging) to isolate the effects of context in making decisions and eliminate the complexity of the learning methods. Specifically, our work makes the following contributions:

- We show through evaluation about the quality to extract reliable semantics in the absence of sentence structure from word representations as independent resources. We discovered difference in the characteristics of the publicly released word embedding according to how they

[2] ICML 2013 Workshop on Deep Learning for Audio, Speech, and Language Processing.

are trained and indicate what features should be emphasized in the optimization function.

- We explore the impact of the number of dimensions and the resolution of each dimension on the quality of the information that can be encoded in the embedding space. Our research shows the redundancy level of word representations and indicates balance between time and quality to capture the useful semantic information in the embedding.

- We demonstrate the importance of relative positions and orientations of embedding pairs in encoding useful linguistic information. We run two pair classification tasks and provide an example with one of them where pair performance greatly exceeds that of individual words.

Figure 2.2: 2-D visualization sample of word representations. Compared with raw formats in Figure 2.1, correct processing including projection and dimension reduction provides much better visualization for human to learn specific knowledge inside word representations.

2.2 Related work

The original work for generating word embedding was presented by Bengio et al. [10] as a secondary output when generating language model. Since then, a significant interest grew in speeding up the generation process [11, 12]. These original language models were evaluated using perplexity.

There had been recent interest in the application of embedding for learning features and representations. SENNA's embedding [25] was generated using a model that is discriminating and non-probabilistic. In each train-

ing update, an n-gram from the corpus was read, concatenating the learned embedding of these n words. Then a corrupted n-gram was used by replacing the word in the middle with a random one from the vocabulary. On top of the two phrases, the model learned a scoring function that scored the original phrases lower than the corrupted one. The loss function used for training was hinge loss. [26] showed that embedding is able to perform well on several NLP tasks in the absence of any other features. The NLP tasks considered by SENNA all consist of sequence labeling, which imply that the model might learn from sequence dependencies. Our work enriches the discussion by focusing on term classification problems.

Turian et al. [94] duplicated the SENNA embedding with some differences; they corrupt the last word of each n-gram instead of the word in the middle. They also showed that using embedding in conjunction with typical NLP features improves the performance on the Named Entity Recognition task. An additional result of [94] showed that most of the embedding has similar effect when added to an existing NLP task. However, this gave the wrong impression. Our work illustrates that not all embedding are created equal and there are significant differences in the information captured by each publicly released model exist.

Mnih and Hinton [71] proposed a log-bilinear loss function to model language. Given an n-gram, the model concatenated the embedding of the n-1 first words, and learned a linear model to predict the embedding of the last word. Mnih and Hinton [72] later proposed Hierarchical log-bilinear (HLBL) embedding to speed up model evaluation during training and testing by using a hierarchical approach (similar to [73]) that prune the search space for the next word by dividing the prediction into a series of predictions that filter region of the space. The language model was eventually evaluated using perplexity.

Huang et al. [45] incorporated global context to handle challenges raised by words with multiple meanings, which was considered a fundamental challenge for neural language models that involves representing words which have multiple meanings.

Mikolov et al. [70] investigated linguistic regularities captured by the relative positions of points in the embedding space, showing that it is possible to find analogous relationship between words (e.g. King : Queen = Man : Woman). Our results regarding pair classification are complementary.

2.3 Experimental setup

We construct three term classification problems and two pair classification problems to measure the quality of embedding.

2.3.1 Evaluation tasks

Our evaluation tasks are as follows:

- **Sentiment polarity**: We use Lydia's sentiment lexicon [40] to create sets of words which have positive or negative connotations and construct the 2-class sentiment polarity test. The data size is 6923 words. We also consider a 3-class version of the sentiment test, in which we discriminate between words that are positive, negative, and neutral. We pick our set of neutral words by randomly selecting from words not occurring in our sentiment lexicon.

- **Noun gender**: We use Bergsma's dataset [13] to compile a list of masculine and feminine proper nouns. Names that co-refer more frequently with *she*/*he* are respectively considered feminine/masculine. Strings that co-refer the most with *it*, appear less than 300 times in the corpus, or consist of multiple words are ignored. The total size is 2133 words.

- **Plurality**: We use WordNet [34] to extract nouns in their singular and plural forms. The data consists of 3012 words.

- **Synonyms and antonyms**: We use WordNet to extract synonym and antonym pairs and check whether we can part one kind from the others. The relation is symmetric thus we put each word pair together with their order-reversed-counterparts. There are 3446 different word pairs. We also consider a 3-class version of this test which adds a new group of containing words that are neither synonyms nor antonyms.

- **Regional spellings**: We collect the words that differ in spelling between UK English and the American counterpart from an online source [63]. We make this task be a pair classification task to emphasize relative distances between corresponding embedding. In total we have 1565 word pairs in this task.

	Sentiment		**Noun Gender**		**Plurality**	
	Positive	Negative	Feminine	Masculine	Plural	Singular
Samples	good	bad	Ada	Steve	cats	cat
	talent	stupid	Irena	Roland	tables	table
	amazing	flaw	Linda	Leonardo	systems	system

	Synonyms and Antonyms		**Regional Spellings**	
	Synonyms	Antonyms	UK first	US first
Samples	store shop	rear front	colour color	color colour
	virgin pure	polite impolite	syphon siphon	siphon syphon
	permit license	friend foe	aeon eon	eon aeon

Table 2.1: Example data input for each task. Top 3 tasks are term-wise comparisons accepting embedding of only one word as input. Bottom 2 tasks are pair-wise comparisons and we feed two word representations at a time.

Notice that the task of "Synonyms and antonyms" should preserve symmetry (i.e. *good* is an antonym of *evil* implies that *evil* is an antonym of *good*) while the "Regional spelling" task is asymmetric.

We ensure that for all tasks the class labels are balanced. This allows our baseline evaluation to be either the random classifier or the most frequent label classifier. Either of them will give an accuracy of 50%. Table 2.1 shows examples of each of the 2-class evaluation tasks. The classifier is asked to identify which of the classes a term or pair belongs to.

2.3.2 Embedding datasets

On the other hand, we choose the following publicly available embedding datasets for evaluation.

- **SENNA's embedding** [25] covers 130,000 words with 50 dimensions for each word.
- **Turian's embedding** [94] covers 268,810 words, each represented either with 25, 50 or 100 dimensions.
- **HLBL's embeddings** [72] covers 246,122 words. This embedding was trained on same data used for Turian embedding for 100 epochs (7 days), and has been induced in 50 or 100 dimensions.

- **Huang's embedding** [45] covers 100,232 words, in 50 dimensions. Huang's embedding requires context to disambiguate which prototype to use for a word. Our tasks are context free so we average the multiple prototypes to a single point in the space, given the fact that this was the approach which worked best in our testing.

It should be emphasized that each of these models has been induced under substantially different training parameters. Each model has its own vocabulary, used a different context size, and was trained for a different number of epochs on its training set. While the control of these variables is outside the scope of this study, we hope to mitigate one of these challenges by running our experiments on the vocabulary shared by all these embedding datasets. The size of this shared vocabulary is 58,411 words.

2.3.3 Classification

For classification we used Logistic regression and a SVM with the RBF-kernel as linear and non-linear classifiers. There is a model-selection procedure by running a grid-search on the parameter space with the help of the development data. All experiments were written using the Python package Scikit-learn [77]. For the term classification tasks we offered the classifier only the embedding of the word as an input. For pairwise experiments, the input consists of the embedding of the two words concatenated.

The average of four folds of cross validation is used to evaluate the performance of each classifier on each task. 50%, 25%, 25% of the data is used, as training, development and testing datasets respectively, for evaluation and model selection.

2.4 Evaluation Results

Here we present the evaluation of both our term and pair classification results.

2.4.1 Term Classification

Figure 2.3 shows the results over all the 2-class term classification tasks using logistic regression and RBF-kernel SVM. It is surprising that all the embedding we considered did much better than the baseline, even on a seemingly

hard test like sentiment detection. What's more, there is strong performance from both the SENNA and Huang embedding. SENNA embedding seems to capture the plurality relationship better, which may be from the emphasis that the SENNA embedding place on shallow syntactic features.

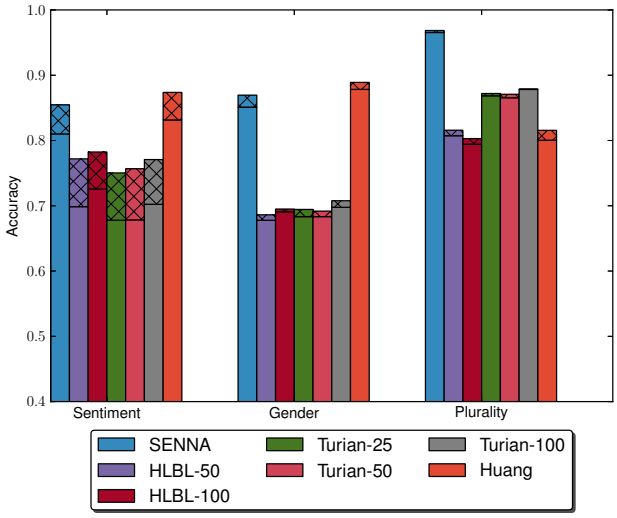


Figure 2.3: Results of term-based tasks. To illustrate that strong performance is still possible on such tasks, we report results by classifier type separately. Unshaded areas show average results from the tasks across classifiers using the geometric mean. Shaded areas represent improvements using kernel SVM.

Table 2.2 shows examples of words from the test datasets after classifying them using logistic regression on the SENNA embedding with most obvious sentiment polarities and British spellings. For SENNA the performance of Sentiment task is good, given the obvious contrast between the probabilities of words – top words are given almost 100% probability and the bottom ones are given almost 0%. The results of regional spelling task shown here use term-wise setup (i.e. judge the regions by the embedding of a single word). Despite not performing as well as the pair-wise spelling (i.e. induce regional spelling by relative positions in embedding space), we can see that classifier shows meaningful results. We can clearly notice that the British spellings of words favor the usage of hyphens, *s* over *z* and *ll* over *l*.

Sentiment			Regional Spelling		
Positive	Prob		**British**	Prob	
world-famous	99.85		kick-off	92.37	
award-winning	99.83		hauliers	91.54	
high-quality	99.83		re-exported	89.46	
achievement	99.81		bullet-proof	88.69	
athletic	99.81		initialled	88.42	
resilient	50.14		paralysed	50.16	
ragged	50.11		italicized	50.04	
discriminating	50.10		exorcise	50.03	
stout	49.97		fusing	49.90	
lose	49.83		lacklustre	49.78	
bored	49.81		subsidizing	49.77	
bloodshed	0.74		signaling	32.04	
burglary	0.68		hemorrhagic	21.69	
robbery	0.58		tumor	21.69	
panic	0.45		homologue	19.53	
stone-throwing	0.28		localize	17.50	
Negative	1.0-Prob		**American**	1.0-Prob	

Table 2.2: Most positive and negative examples using logistic regression on Sentiment task and Regional spelling task. *Prob* column measures how well the example fit into one category. Word like *resilient* could have positive and negative connotations in text and we find it close to the region were the words are more neutral than being polarized.

2.4.2 Pair Classification

Sometimes however, the choice to use pair classification can make quite a difference in the results. Figure 2.4 shows that classifying individual words according to their regional usage performs poorly while redefining the problem to decide if the first word, in a pair of words, is the American spelling or not improves the performance improves a lot. Such phenomenon indicates that some words pairs are not separable by a hyper-plane in any subspace of the original embedding space. Instead, we draw a similar conclusion as [70] that the pairs' positions relative to each other is what encodes such information but not their absolute coordinates, and relationship between words often indicate the relative difference vector between corresponding points.

In order to show how well linguistic information is encoded in the embed-

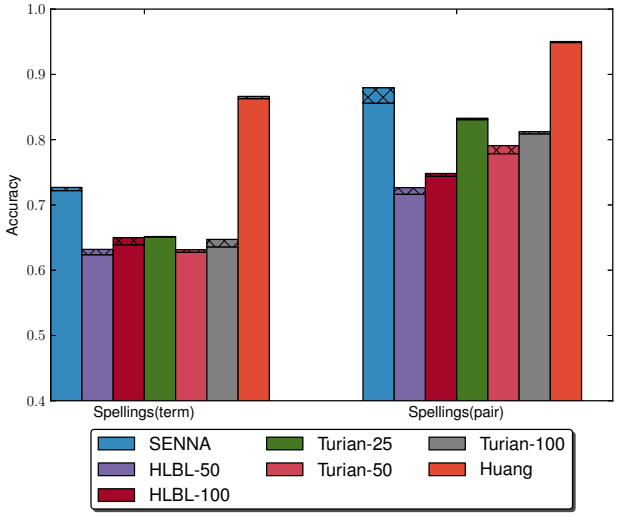


Figure 2.4: Comparison between treating Region spelling task as term-based test and as pair-based test. The result shows that difference between embedding encoded more information than absolute positions in embedding space.

ding of word pairs, we present the results of our 2-class pair tasks in Figure 2.5. The embedding performs well as expected. Plus, an interesting difference between SENNA and Huang's embedding can be observed here. In our previous Plurality test, the SENNA embedding significantly outperformed Huang's. However in our regional spelling task (which might seem similar), Huang's embedding outperform SENNA in both term and pair classification setups. We believe that Huang's approach for building word prototypes from significant differences in context provide a significant advantage on this task.

Given the fact that they way both HLBL and SENNA/Turian model corrupted their examples favor words that can syntactically replace each other; e.g. *bad* can replace *good* as easily as *excellent* can. The result of this syntactic interchangeability is that both *bad* and *excellent* are close to *good* in the embedding space. However, it is good to see these models may capture the relation between a synonym and antonym well, indicating that minor differences of context around synonyms and antonyms are captured during the training.

In general, Huang's embedding performed best on the 2-class tests. The notable exception was the Plurality task, which was the strongest performing

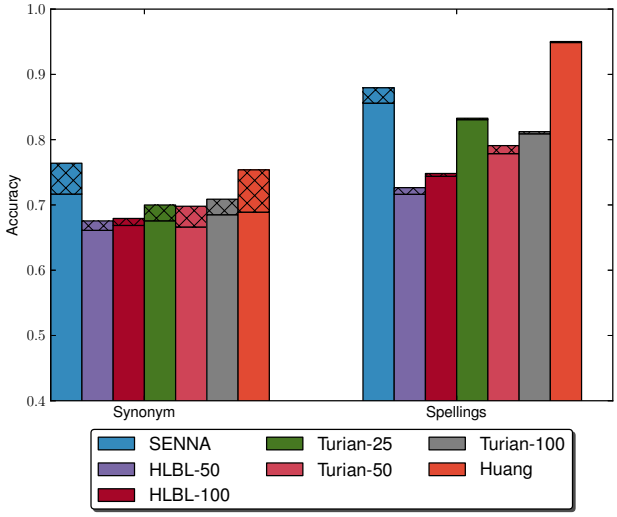


Figure 2.5: Results of pair-based tests on 2-class classification tasks. Unshaded areas show average results from the tasks across classifiers using the geometric mean. Shaded areas represent improvements using kernel SVM.

task for each of the other embedding. Huang's embedding seeks to primarily capture semantic relationships. SENNA also performed strongly on the 2-class tests. On the 3-class tests, SENNA performs better than Huang's embedding.

To better explain these results, we performed a 3-class version of the sentiment test, in which we evaluated the ability to classify words as having positive, negative, or neutral sentiment value. The results are presented in Figure 2.6. In order to show that embedding can still perform quite well on this task, we have reported the nonlinear classifier separately from the linear ones. The results are consistent with those from our 2-label test, and all embedding performs much higher than the baseline score of 33%.

Besides, we conducted a similar 3-class version test on Synonyms and antonyms task to investigate the depth to which semantic features are captured. We now evaluate between pairs of words that are synonyms, antonyms, or have no such relation. While such a task is much harder for the embedding, the results in Figure 2.7 show that a nonlinear classifier can capture the relationship, particularly with the SENNA embedding. An analysis of the confusion matrix for the nonlinear SVM showed that errors occurred roughly

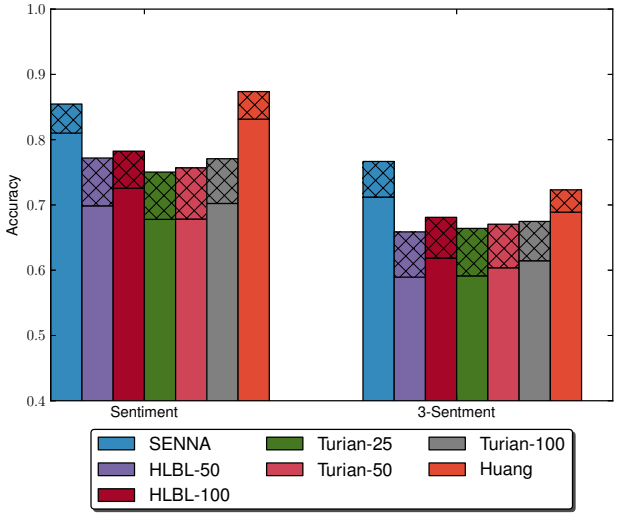


Figure 2.6: Performances on the 3-class version of Sentiment task. Unshaded areas show average results from the tasks across classifiers using the geometric mean. Shaded areas represent improvements using kernel SVM.

evenly between the classes. We believe that this finding regarding the encoding of synonym/antonym relationships is an interesting contribution of our work.

2.5 Information reduction

Distributed word representations exist in continuous space, which is quite different from common language modeling techniques. Beside the powerful expressiveness that we demonstrated previously, another key advantage of distributed representations is their size - they require far less memory and disk storage than other techniques. In this section we seek to understand exactly how much space word embedding need in order to serve as useful features. We also investigate whether the powerful representation that embedding offer is a result of having real value coordinates or the exponential number of regions which can be described using multiple independent dimensions.

To understand the effect of such hyper-parameters we run two experiments. The first reduces the resolution of each real-valued dimension and

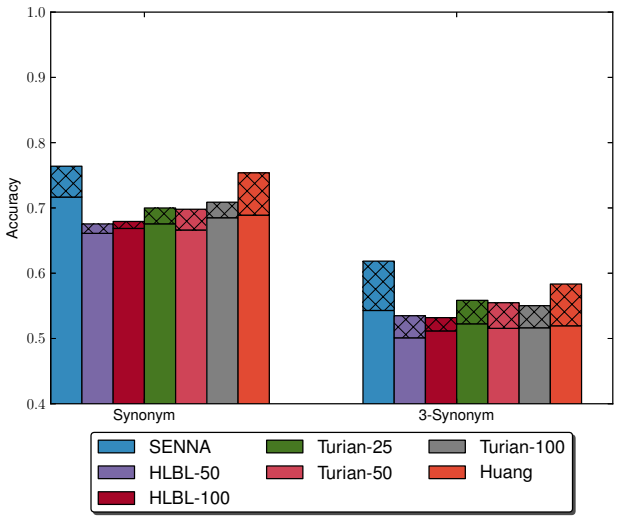


Figure 2.7: Performances on the 3-class version of Synonyms and antonyms task. Unshaded areas show average results from the tasks across classifiers using the geometric mean. Shaded areas represent improvements using kernel SVM.

helps us understand the level of precision required for our tasks. The second reduces the dimensions of embedding and provides insight into how the dimensions of the embedding affect the final result.

2.5.1 Bitwise truncation

To reduce the resolution of the real numbers those make up the embedding matrix. First we scale them to 32 bit integer values, then we divide the values by 2^b, where b is the number of bits we wish to remove. Finally, we scale the values back to lie between $(-1, 1)$. After this preprocessing we give the new values as features to our classifiers. In the extreme case, when we truncate 31 bits, the values will be all either $\{1, -1\}$.

Figure 2.8 shows that when we remove 31 bits (i.e, values are $\{1, -1\}$), the performance of an embedding dataset drops no more than 7%. This reduced resolution is equivalent to 2^{50} regions which can be encoded in the new space. This is still a high resolution, but surprisingly seems to be sufficient at solving the tasks we proposed. A naïve approximation of this trick which

may be of interest is to simply take the sign of the embedding values as the representation of the embedding themselves. Figure 2.9 illustrate changes of performance in different tasks. All tasks behave similarly and the most distinguished threshold of resolution lies near 4 bits.

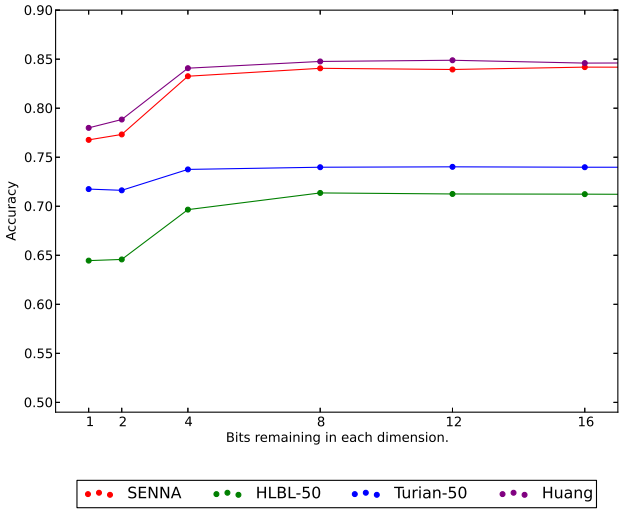


Figure 2.8: Results of reducing the precision of the embedding, averaged by the geometric mean of each dataset. Notice that after removing 31 bits, each dimension of the embedding is a binary feature.

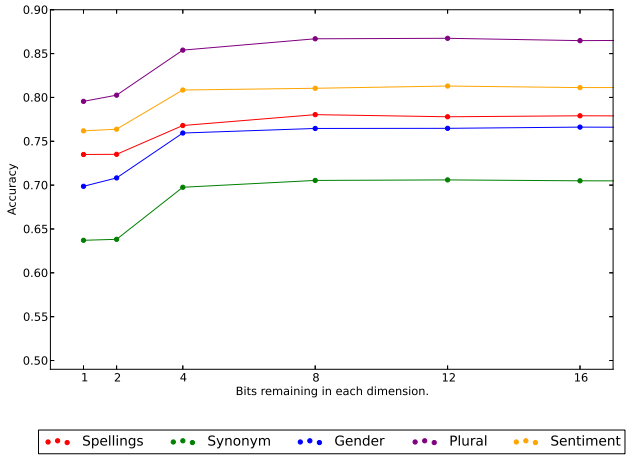


Figure 2.9: Results of reducing the precision of the embedding, averaged by the geometric mean of tasks.

2.5.2 Principle component analysis (PCA)

The bitwise truncation experiment indicates that the number of dimensions could be a key factor into the performance of the embedding. To experiment on this further, we run PCA over the embedding datasets to evaluate task performance on a reduced number of dimensions. Figure 2.10 shows that reducing the dimensions drops the accuracy of the classifiers significantly across all embedding datasets and Figure 2.11 shows that such behavior exist for all tasks. It is expected since the dimensionality reduction is unsupervised but the slope is still an interesting point to discover.

If subtle semantic features such as sentiment polarity need more dimensions to explain, shallow syntactic features such as gender and number agreement may be preserved at the expense dimension redundancy. This gives us insight into what the hierarchical structure of the embedding space looks like. Shallow semantic features are present in all aspects of the space, and when PCA chooses to maximize this variance of the feature space it is at the expense of the other semantic properties.

Another key difference between the truncation experiment and the PCA experiment is that the truncation experiment may preserve relationships captured by non-linearities in the embedding space. Linear PCA cannot offer such guarantees and this weakness may contribute to the difference in per-

formance.

Looking at Figure 2.11, reducing the words embedding to points on a real line almost deletes all the features that are relevant to the pair classification and to less a degree the sentiment features. Despite the 10%-20% drop in accuracy in the Plurality and Gender tasks, the classification is still higher than random. The results shows shallow syntactic features such as gender and number agreement are preserved at the expense of more subtle semantic features such as sentiment polarity. This gives us insight into what the hierarchical structure of the embedding space looks like. Shallow semantic features are present in all aspects of the space, and when PCA chooses to maximize this variance of the feature space it is at the expense of the other semantic properties.

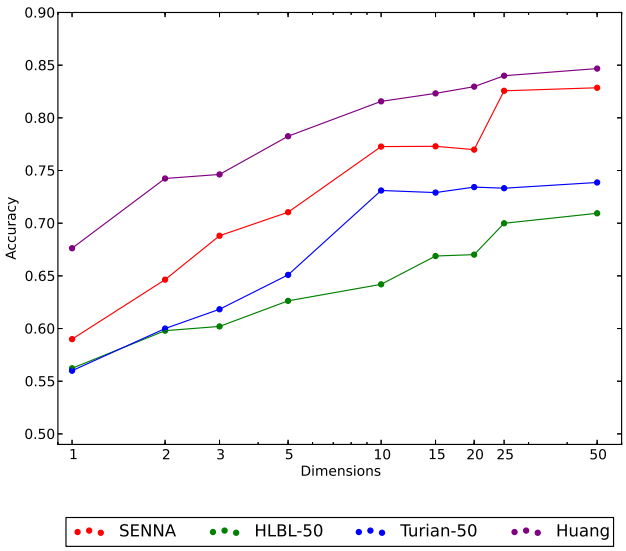


Figure 2.10: Results of reducing the dimensions of the embeddings through PCA, averaged by the geometric mean of each dataset.

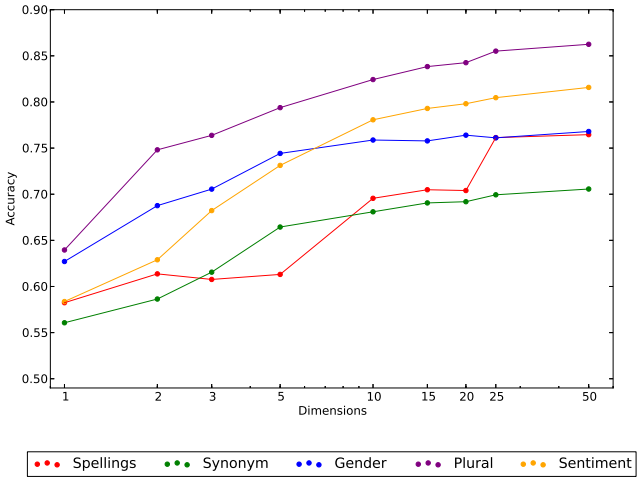


Figure 2.11: Results of reducing the dimensions of the embeddings through PCA, averaged by the geometric mean of each task.

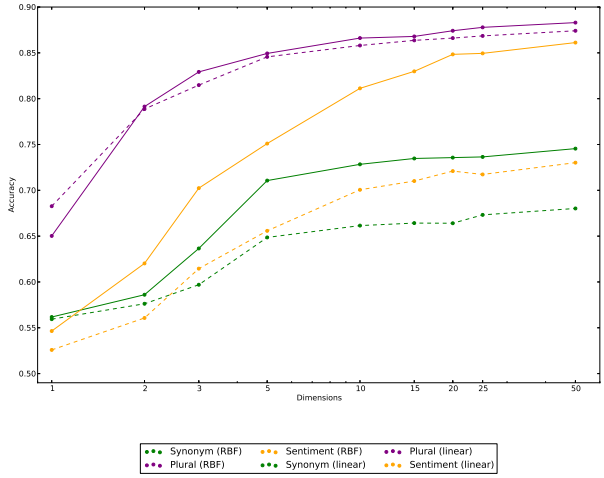


Figure 2.12: Comparison of performance changes between linear classifiers and nonlinear classifiers during PCA.

We also illustrate this phenomenon in Figure 2.12 by showing how performance of the linear and non-linear classifiers converge for harder tasks (sentiment and synonym) as we reduce the number of dimensions in PCA.

2.6 Application of word representations

Previous work on distributed word representations mainly focused on speeding up the training process with one metric for evaluation, perplexity. We show that this metric is not able to provide a nuanced view of their quality. We develop a suite of linguistic oriented tasks which might serve as a part of a comprehensive benchmark for word embedding evaluation. The tasks focus on words or pairs of them in isolation to the actual text. The goal here is not to build a useful classifier as much as it is to understand how much supervised learning can benefit from the features encoded in the embedding.

Word representations show a lot of promise to improve supervised learning and semi-supervised learning. As single-word features that could be integrated in to Word Connection Networks , word representations are proved to be valuable on many natural language processing tasks. Having dense representations within groups of similar words provides a collection of unsupervised language features from context that benefit many language processing tasks, given the fact that information extracted from word representations basically agree with human knowledge stored in Word Connection Networks.

We succeed in showing that the publicly available datasets differ in their quality and usefulness, and our results are consistent across tasks and classifiers. Our future work will try to address the factors that lead to such diverse quality. The effect of training corpus size and the choice of the objective functions are two main areas where better understanding is needed.

While our tasks are simple, the differences among task performance shed light on the features encoded by embedding. We showed that in addition to the shallow syntactic features like plural and gender agreement, there are significant semantic partitions regarding sentiment and synonym/antonym meaning. On the other hand, we demonstrate high redundancy in word representations. Developing better approaches of training procedure/storing mechanism/information retrieval will potentially improve the performance of word representations on many natural language processing tasks.

Last but not least, word representations demonstrate an innovative idea of enriching intra-language connections inside Word Connection Networks. Multiple trustful relationships between words in each single language could be extracted via specific distance function trained by machine learning methods. Combined with inter-language connections across languages, we can capture more interesting features in multilingual environment that are not displayed from original data collections in Word Connection Networks.

Chapter 3

Constructing Multilingual Sentiment Lexicons

Sentiment analysis of English texts has become a large and active research area, with many commercial applications including market research, opinion polling, and product review analysis. But the barrier of language limits the ability to assess the sentiment of most of the world's population.

Although several well-regarded sentiment lexicons are available in English [33, 64], the same is not true for most of the world's languages. Indeed, our literature search identified only 12 publicly available sentiment lexicons for only 5 non-English languages (Chinese mandarin, German, Arabic, Japanese and Italian). No doubt we missed some, but it is clear that these resources are not widely available for most important languages.

Sentiment lexicons alone can tell enough information and produce comprehensive set of sentiments for in multilingual environment, though grammar or sentence based sentiment analysis is definitely necessary to achieve robust performance in different languages, including applying language specific negation words, amplifiers and sentence structures. We address this lexicon gap by building high-quality sentiment lexicons through graph propagation method in our Word Connection Networks for 136 world's major languages. We believe high-coverage of trustful sentiment lexicons in major languages would greatly benefit natural language processing tasks in multilingual world.

3.1 Our work

We strive to evaluate transitivity and conflicts of connections in Word Connection Networks and produce a comprehensive set of sentiment lexicons for the worlds' major languages. We make the following contributions in our work [22]:

- **New sentiment analysis resources** – We have generated sentiment lexicons for 136 major languages via graph propagation. We validate our own work through other publicly available, human annotated sentiment lexicons. Indeed, our lexicons have polarity agreement of 95.7% with these published lexicons, plus an overall coverage of 45.2%.

- **Large-scale language knowledge graph** – We have created a massive comprehensive Word Connection Networks of 7 million vocabulary words from 136 languages with over 131 million semantic inter-language links. This graph has highly heterogeneous types of edges which proves valuable when doing alignment between definitions in different languages.

- **Graph analysis and sentiment passing algorithms** – We experimented different graph propagation algorithms on Word Connection Networks. We perform experiments to evaluate the importance of each class of language resources to our final comprehensive sentiment with a grid search to find best confidence level of each type of connections. We successfully resolve potential conflictions of translations using different bridge languages and pass sentiment lexicons to many languages without such resources.

- **Extrinsic Evaluation** – Since we do not have comparable sentiment lexicons in all languages, we conduct an experiment to check if our propagated sentiment lexicons can preserve relative opinions from Wikipedia pages in different languages, assuming that Wikipedia keep their text in a neutral point of view. In particular, we elucidate the sentiment consistency of entities reported in different language editions of Wikipedia using our propagated lexicons compute sentiment scores for 2,000 distinct famous historical figures. Language pairs among biggest 30 Wikipedia languages exhibits a minimum Spearman sentiment correlation of 0.14 and an average correlation of 0.28.

3.2 Related work

Sentiment analysis is an important area of NLP with a large and growing literature. Excellent surveys of the field include [65, 76, 44], establishing that rich online resources have greatly expanded opportunities for opinion mining and sentiment analysis. Godbole et al. [40] built an English lexicon-based sentiment analysis system to evaluate the general reputation of entities. Taboada et al. [93] presented a more sophisticated model by considering patterns, including negation and repetition using adjusted weights. Liu [64] introduced an efficient method, at the state of the art, for doing sentiment analysis and subjectivity in English.

Researchers had investigated topic or domain dependent approaches to identify opinions. Jijkoun et al. [49] focused on generating topic specific sentiment lexicons. Li et al. [62] extracted sentiment with global and local topic dependency. Gindl et al. [38] performed sentiment analysis according to cross-domain contextualization and Pak and Paroubek [75] focused on Twitter, doing research on colloquial format of English.

Work had been done to generalize sentiment analysis to other languages. Denecke [29] performed multilingual sentiment analysis using SentiWordNet. Mihalcea et al. [69] learned multilingual subjectivity via cross-lingual projections. Abbasi et al. [1] provided a method of extracting specific language features of Arabic, which requires language-specific expertise. Ahmad et al. [4] demonstrated that their approach can basically handle sentiment words in the financial arena, but making extension based on their approach requires substantial human effort. There were other language-dependent efforts [8, 50, 84, 2, 9, 42, 39, 41], all attempting to produce better sentiment lexicons or sentiment analysis system in foreign languages.

The ready availability of machine translation to and from English had prompted efforts to employ translation for sentiment analysis [9]. Banea et al. [7] demonstrated that machine translation can perform quite well when extending the subjectivity analysis to multi-lingual environment, which makes it inspiring to replicate their work on lexicon-based sentiment analysis.

Machine learning approaches to sentiment analysis are attractive, because of the promise of reduced manual processing. Boiy and Moens [16] conducted machine learning sentiment analysis using multilingual web texts. Deep learning approaches drafted off of distributed word embedding which offer concise features reflecting the semantics of the underlying vocabulary. Turian et al. [94] created powerful word embedding by training on real and

corrupted phrases, optimizing for the replaceability of words. Zou et al. [105] combined machine translation and word representation to generate bilingual language resources. Socher et al. [90] demonstrated a powerful approach to English sentiment using word embedding, which can easily be extended to other languages by training on appropriate text corpora and benefitted language processing tasks in foreign languages.

3.3 Resource Statistics

3.3.1 Statistics of Word Connection Networks

We collect 5 types of connections that are useful in passing sentiment messages, including translations from Google, Wiktionary reference links, exact script matching, synonyms and antonyms, covering 100,000 most frequent Wikipedia words in each of 136 languages. We here provide basic statistics of the Word Connection Networks in creating sentiment lexicons in Table 3.1.

Features	**Count**
Languages	136
Vertices	7,741,544
Isolated vertices	1,732,755
Directed edges	131,773,405
Wiktionary connections	1,315,755
Script matching	66,474,432
Google Translation	23,866,910
Synonyms	97,664
Antonyms	2,754

Table 3.1: Statistics of Word Connection Networks. *Isolated vertices* are vertices without any semantic links in Word Connection Networks. *Synonyms edges* and *Antonyms edges* only consider those in English from *SentiWordNet*.

Figure 3.1 and Table 3.2 demonstrate links statistics in 10 biggest Wikipedia families. Even though these languages have relatively good connectivity to the rest of the network, typically we have 5% to 20% of the words disconnected from the rest of the graph. Such phenomena demonstrates the

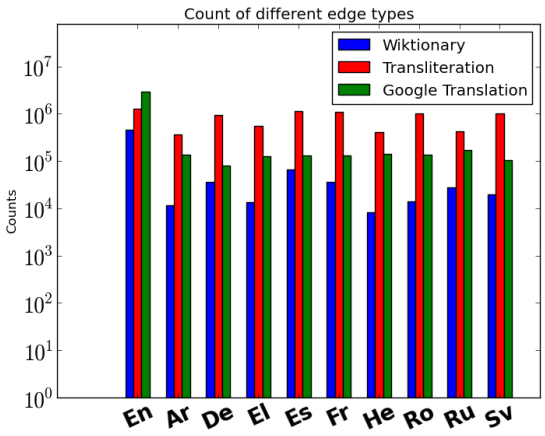


Figure 3.1: Count of different edges represented in log format of 10 biggest Wikipedia languages. English acts as the hub of all languages thus it contains many google translation links. Transliteration links provides word pairs with spelling and sound similarity. Wiktionary links provides semantic connections that are not discovered by Google translations.

Lang Code	**Isolated Vertices**	**Avg Deg.**	**Wiki Deg.**	**Script Deg.**	**Google Deg.**
Ar	16.77%	1.46	0.12	3.69	1.39
De	20.18%	1.05	0.37	9.58	0.79
El	14.43%	1.33	0.14	5.65	1.25
En	5.11%	31.28	4.50	12.59	28.90
Es	4.39%	1.86	0.66	11.36	1.34
Fr	6.24%	1.56	0.36	10.91	1.32
He	14.52%	1.17	0.07	3.25	1.13
Ro	8.19%	1.44	0.14	10.02	1.36
Ru	4.75%	1.89	0.28	4.28	1.71
Sv	15.33%	1.19	0.20	10.04	1.06

Table 3.2: Network statistics of 10 biggest Wikipedia languages. We calculated fraction of isolated vertices as well as average vertex degrees by edge source to give a brief idea of the structure of Word Connection Networks. English works as a hub in collecting translations from and to other languages so it has a great number of average degrees.

importance of selecting vocabularies according to tasks to avoid waste of computational power.

3.4 Graph Propagation

Sentiment propagation starts from the polarities of a root English sentiment lexicon. Through semantic links in our knowledge graph, words are able to extend their sentiment polarities to their neighbors. We experimented with both belief propagation algorithm [98] and label propagation algorithm [104, 82]. Differences between two propagation methods are shown in Figure 3.2 that label propagation takes all paths from seed node to target vertex into consideration, resolving conflictions multiple times, while belief propagation utilizes only the best path between seed node and target vertex when resolving conflictions.

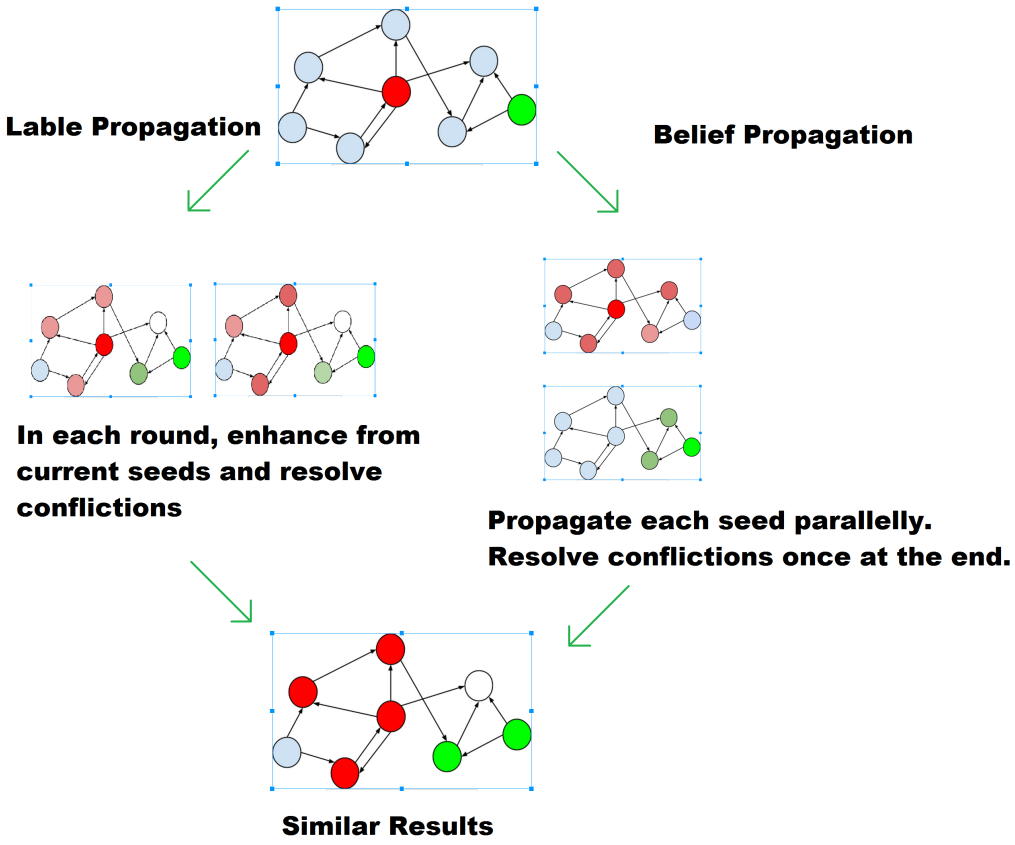


Figure 3.2: Illustration of label propagation and belief propagation. Initial Red / Green nodes represent positive / negative seed words. Blank nodes will be affected via links during propagation, gaining sentiment of close neighbors. Label propagation resolves conflictions after each round belief propagation record only the best path in parallel and resolve conflictions once at the end.

In detail, the belief propagation algorithm proceeds as shown in Algorithm 1:

Algorithm 1 Sentiment propagation algorithm

Input: Graph G = (V, E), $w_{ij} \in [0, 1]$
P as Positive seeds, N as Negative seeds
Output: Pol_i for each vertex i
Initialize: Pol_i, Pol_i^+, $Pol_i^- = 0$ for all i
1. Set $\alpha_{ij}^+ = \alpha_{ij}^- = 0$ for all i, j
2. For $v_i \in P$:
3. F = $\{v_i\}$
4. If found F extendable:
5. For $(v_k, v_j) \in E$ where $v_k \in F$
6. $\alpha_{ij}^+ = max\{\alpha_{ij}^+,\ \alpha_{ik} \cdot w_{kj}\}$
$\alpha_{ij}^- = min\{\alpha_{ij}^-,\ \alpha_{ik} \cdot w_{kj}\}$
F = F $\cup\{v_j\}$
7. For $v_j \in V$:
$Pol_i^+ = \Sigma_{v_i \in P}\ \alpha_{ij}^+$
$Pol_i^- = \Sigma_{v_i \in P}\ \alpha_{ij}^-$
8. Repeat 1-7 using N to compute Pol^-
9. Set $\beta = \Sigma_i Pol_i^+ \div \Sigma_i Pol_i^-$
10. $Pol_i = Pol_i^+$ - βPol_i^-

The variable T controls the max path length considered by the algorithm. In practice, we found that restricting T to at most 1 or 2 hops is both more efficient and reliable, since long paths from seeds rarely contribute to accurate polarity scores. The variable w controls the confidence level of different types of edges; for example we anticipate that transliteration edges may be less accurate or reliable than vetted Wiktionary definitions. We distinguish both α^+ and α^- because antonym word pairs lead to polarity reversal from seed words in our algorithm.

3.4.1 Why not Label Propagation?

Certain previous studies [82] on constructing polarity lexicons use the label propagation algorithm described in [104]. Label propagation is an iterative algorithm where each vertex takes on weighted average of its neighbor's values from previous iteration.

Detailed comparison of these two propagation results can be found in next section, but in summary our propagation algorithms perform slightly

better than label propagation algorithm. Plus, since our Word Connection Networks is a sparse graph, there are no needs to resolve conflicts after each round of propagations. Considering only the best path will save processing time and provide better summary for propagation results of each seed word.

3.4.2 Source English Sentiment Lexicons

We report results from using Liu's lexicons [64] as seed words. Liu's lexicons contain 2006 positive words and 4783 negative words. Of these, 1422 positive words and 2956 negative words (roughly 64.5%) appear among the 100,000 English vertices in our graph.

3.4.3 Parameterization of Edge Classes

Our knowledge network is comprised of links from a heterogeneous collection of sources, of different coverage and reliability, and hence representatives of each class presumably should not be weighted equally. An edge gains zero weight if both negative and positive links exist. For edges defined by multiple source classes, we use the maximum of the possible parameter values. We conducted a grid search for optimal weight values for each connection type, including Machine translation, Wiktionary, Script matching, WordNet synonyms and antonyms, within the range of [-1.0, 1.0] with a step-length of 0.1. To avoid potential over fitting problem, grid search starts from SentiWordNet English lexicons [33] instead of Liu's. Our objective function maximizes the overall agreement (e.g. accuracy) on our Liu's dataset of published non-English sentiment lexicons.

Table 3.3 shows some sample edge-weight sets and we hope this table can tell more about the quality of our resources.

Our experiment demonstrates that Google translation and Wiktionary resources provide the most reliable semantic links in extending sentiment. Transliteration links prove very valuable to increase the connectivity of our graph – particularly since we trust only short paths from seed words but there are sometimes flip of sentiment polarity via transliteration links. Synonym links prove less trustworthy than antonyms, primarily because due to existence of multi-sense words, which often confuse sentiment propagation.

Type	**Best**	-Ant	-Syn	-Google	-Script	-Wik
Wiktionary	**0.8**	1.0	1.0	1.0	1.0	0.1
Script matching	**0.5**	1.0	1.0	1.0	0.1	1.0
Google Translation	**0.8**	1.0	1.0	0.1	1.0	1.0
Synonyms	**0.7**	1.0	0.1	1.0	1.0	1.0
Antonyms	**-0.7**	-0.1	-1.0	-1.0	-1.0	-1.0
Accuracy Performance	**0.94**	0.85	0.89	0.78	0.85	0.82

Table 3.3: Edge parameter weights and corresponding performances. *Best* column demonstrates the optimized parameters via grid search. Following columns each shows values and performances of weakening a certain type of edges. Performances are measured via the agreement of our test dataset of published non-English sentiment lexicons.

3.5 Lexicon Evaluation

We collected all available published sentiment lexicons from non-English languages to serve as standard for our evaluation, including Arabic, Italian, Deutsch and Chinese. Coupled with English sentiment lexicons provides in total seven different test cases to experiment against, specifically:

- **Arabic**: Wordlist in subjectivity analysis [2].
- **German**: Generated German language resources in [84].
- **English**: SentiWordNet [33].
- **Italian**: Sentiment lexicons in Italian tweets [8].
- **Japanese**: Collected from massive HTML documents [50].
- **Chinese-1**, **Chinese-2**: Chinese sentiment lexicons [41].

3.5.1 Testing Score

Given the fact that we are trying to do a generalized work in multi-lingual environment and our dictionary is created from high-frequency words in each dependent language, it is hard to tell if our lexicons cover a wider range compared with those published sentiment lexicons. However, we could check how

well our propagated sentiment lexicons agree with these previously defined lexicons. We present the accuracy and F1 score achieved by each algorithm variant on each text lexicon in the table below to evaluate the performance of our propagation method.

Accuracy represents the ratio of correct sentiment words (identical polarity between our analysis and the published lexicons). **F1** shows the combination of precision and recall. Finally, **Coverage** reflects what fraction of lexicons reflected by the published lexicon was correctly recovered by our algorithm.

Test	Propagation	Accuracy	F1	Coverage
Ar	Label	0.93	0.93	0.45
	Belief	**0.94**	**0.94**	**0.46**
De	Label	0.97	0.97	0.31
	Belief	**0.97**	**0.97**	**0.32**
En	Label	**0.92**	**0.94**	0.55
	Belief	0.90	0.92	**0.69**
It	Label	**0.73**	**0.77**	0.29
	Belief	0.72	0.76	**0.32**
Ja	Label	**0.57**	**0.72**	0.12
	Belief	0.56	0.71	**0.15**
Zh-1	Label	**0.95**	0.94	0.62
	Belief	0.94	**0.94**	**0.65**
Zh-2	Label	0.97	0.97	0.70
	Belief	**0.97**	**0.97**	**0.72**

Table 3.4: Belief propagation vs label propagation, using Liu's lexicons as seed words. **Accuracy** represents the ratio of identical polarity between our analysis and the published lexicons. **F1** combines precision and recall. **Coverage** reflects what faction of our lexicons overlap with published lexicons.

Comparing the two algorithms (belief vs. label propagation) in Table 3.4, we see that belief propagation always yields better coverage, meaning that we have less confusion on contradicting inputs. The situations where it loses reflect minor losses of accuracy and F1 score. We see similar results in Table 3.5, where we begin from SentiWordNet. Again, belief propagation performs better than label propagation. Finally, we conclude that Liu's lexicons pro-

vide substantially higher coverage and greater accuracy than SentiWordNet.

Test	Propagation	Accuracy	F1	Coverage
Ar	Label	0.87	0.87	0.26
	Belief	**0.88**	**0.87**	**0.30**
De	Label	**0.89**	0.91	0.14
	Belief	0.88	**0.91**	**0.19**
En	Label	0.93	0.95	0.17
	Belief	**0.94**	**0.95**	**0.20**
It	Label	0.78	0.82	0.14
	Belief	**0.80**	**0.83**	**0.18**
Ja	Label	0.40	0.57	0.08
	Belief	**0.42**	**0.61**	**0.14**
Zh-1	Label	**0.86**	**0.86**	0.33
	Belief	0.85	0.85	**0.37**
Zh-2	Label	0.89	0.91	0.29
	Belief	**0.89**	**0.91**	**0.31**

Table 3.5: Belief propagation vs label propagation, starting from SentiWordNet. Measure metrics have the same definition as Table 3.4.

Also, we draw a conclusion that starting from Bin Liu's opinion lexicons is better – simply because that this list contains more words with less confusion.

3.5.2 Per-language Analysis

It is important to compare the effectiveness of our propagation for each of the 136 languages in our analysis, where many of these languages are particular resource poor with respect to text volume and language links. Statistics on all of sentiment lexicons are presented in Table 3.6, including the size of our lexicons and the ratio of words labeled as of positive vs. negative sentiment in each language. It reveals that very sparse sentiment lexicons resulted for a small but notable fraction of the languages we analyzed.

In particular, only 21 languages yielded lexicons of less than 100 words. Accumulated distribution of sentiment word count is shown in Figure 3.3. Without exception, these are language with very small available definitions in Wiktionary mapping indigenous words to other languages as shown in Figure

Language	Count	Ratio	Language	Count	Ratio	Language	Count	Ratio
Afrikaans	2299	0.40	Albanian	2076	0.41	Amharic	46	0.63
Arabic	2794	0.41	Aragonese	97	0.47	Armenian	1657	0.43
Assamese	493	0.49	Azerbaijani	1979	0.41	Bashkir	19	0.63
Basque	1979	0.40	Belarusian	1526	0.43	Bengali	2393	0.42
Bosnian	2020	0.42	Breton	184	0.42	Bulgarian	2847	0.40
Burmese	461	0.48	Catalan	3204	0.37	Cebuano	56	0.54
Chechen	26	0.65	Chinese	3828	0.34	Chuvash	17	0.76
Croatian	2208	0.40	Czech	2599	0.41	Danish	3340	0.38
Divehi	67	0.67	Dutch	3976	0.38	English	4376	0.32
Esperanto	2604	0.40	Estonian	2105	0.41	Faroese	123	0.43
Finnish	3295	0.40	French	4653	0.35	Frisian	224	0.43
Gaelic	345	0.50	Galician	2714	0.37	German	3974	0.38
Georgian	2202	0.40	Greek	2703	0.39	Gujarati	2145	0.44
Haitian	472	0.44	Hebrew	2533	0.36	Hindi	3640	0.39
Hungarian	3522	0.38	Icelandic	1770	0.40	Ido	183	0.49
Interlingua	326	0.50	Indonesian	2900	0.37	Italian	4491	0.36
Irish	1073	0.45	Japanese	1017	0.39	Javanese	168	0.51
Kazakh	81	0.65	Kannada	2173	0.42	Kirghiz	246	0.49
Khmer	956	0.49	Korean	2118	0.42	Kurdish	145	0.48
Latin	2033	0.46	Latvian	1938	0.42	Limburgish	93	0.46
Lithuanian	2190	0.41	Luxembourg	224	0.52	Macedonian	2965	0.39
Malagasy	48	0.54	Malayalam	393	0.50	Malay	2934	0.39
Maltese	863	0.50	Marathi	1825	0.48	Manx	90	0.51
Mongolian	130	0.52	Nepali	504	0.49	Norwegian	3089	0.37
Nynorsk	1894	0.39	Occitan	429	0.40	Oriya	360	0.51
Ossetic	12	0.67	Panjabi	79	0.63	Pashto	198	0.50
Persian	2477	0.39	Polish	3533	0.39	Portuguese	3953	0.35
Quechua	47	0.55	Romansh	116	0.48	Romanian	3329	0.39
Russian	2914	0.43	Sanskrit	178	0.59	Sami	24	0.71
Serbian	2034	0.41	Sinhala	1122	0.43	Slovak	2428	0.43
Slovene	2244	0.42	Spanish	4275	0.36	Sundanese	476	0.50
Swahili	1314	0.42	Swedish	3722	0.39	Tamil	2057	0.40
Tagalog	1858	0.44	Tajik	97	0.62	Tatar	76	0.50
Telugu	2523	0.41	Thai	1279	0.51	Tibetan	24	0.63
Turkmen	78	0.56	Turkish	2500	0.39	Uighur	18	0.44
Ukrainian	2827	0.41	Urdu	1347	0.39	Uzbek	111	0.57
Vietnamese	1016	0.38	Volapuk	43	0.70	Walloon	193	0.32
Waray	44	0.61	Welsh	1647	0.42	Yiddish	395	0.43
Yoruba	276	0.50						

Table 3.6: Statistics of propagated sentiment lexicons in major languages. We tag 10 languages having most/least sentiment words with blue/green color and 10 languages having highest/lowest ratio of positive words with orange/purple color. **Count** shows number of propagated sentiment lexicons and **Ratio** demonstrates the ratio of positive lexicons.

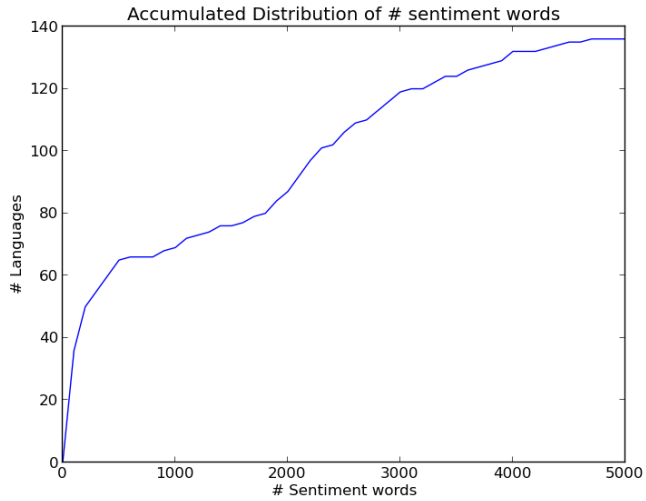


Figure 3.3: Accumulated distribution of sentiment word count. About 20 languages have less than 100 propagated sentiment words. Totally 60 languages have less than 1,000 sentiment words.

3.4, the correlation between Wiktionary entries and sentiment word count. By contrast, 48 languages had lexicons with over 2,000 words, another 16 with between 1,000 and 2,000: clearly large enough to perform a meaningful analysis.

Further find that ratio of positive sentiment words is strongly connected with number of sentiment words. Interestingly, the lexicon seems to become more positive as it becomes smaller, possibly reflecting the fact that many negative words reflect cultural nuances which do not translate well, like *shmuck*, *schlimeil*, and *putz* in Yiddish. We believe that this ratio can be considered as quality measurement for the success of the propagation. It is noteworthy that English has the smallest ratio of positive lexicon terms.

3.6 Extrinsic Evaluation: Consistency of Wikipedia Sentiment

To provide an extrinsic evaluation of our sentiment lexicons, we consider the consistency of evaluation of different language Wikipedia pages about a

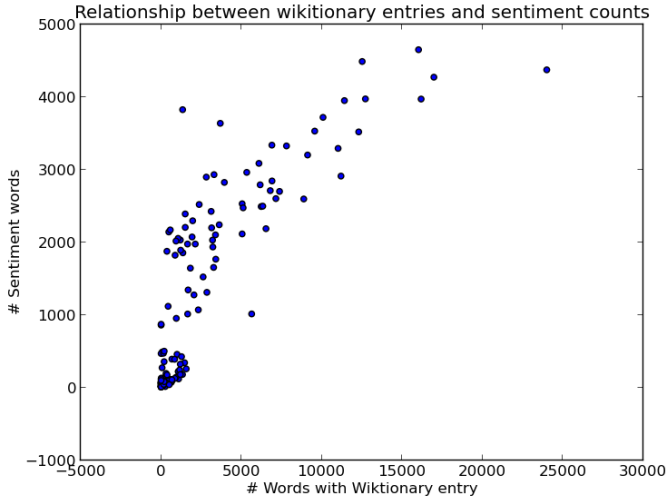
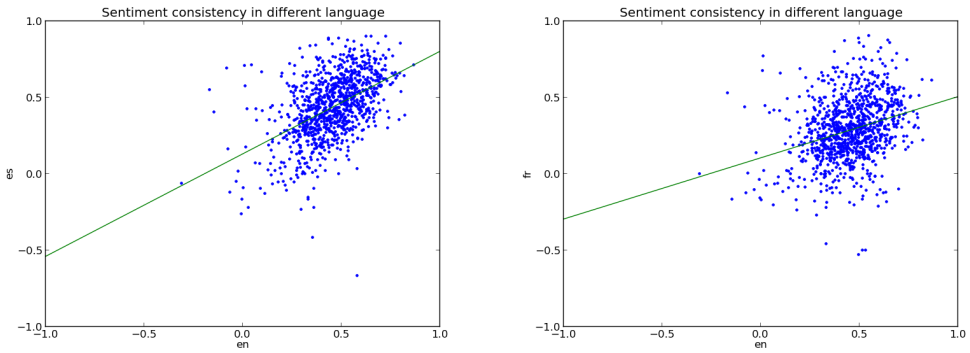


Figure 3.4: Correlation between Wiktionary entries and sentiment word count. Languages with less than 5,000 Wiktionary entries are usually those we cannot find good propagations of sentiment lexicons, indicating that better language resources are needed for these languages.

particular entity, particularly individual people. We assert that the sentiment scale from "evil" to "hero" should show gross consistencies across cultures, although there will also be considerable cultural variation as well. As our candidate entities for analysis, we use the Wikipedia pages of 2,000 most famous people according to their significance as measured in the recent book *Who's bigger?* [88]. Sentiment polarity for a page is simply computed by the number of occurrences of positive polarity words - negative polarity words, divided by the sum of both.

3.6.1 Normalizing across languages

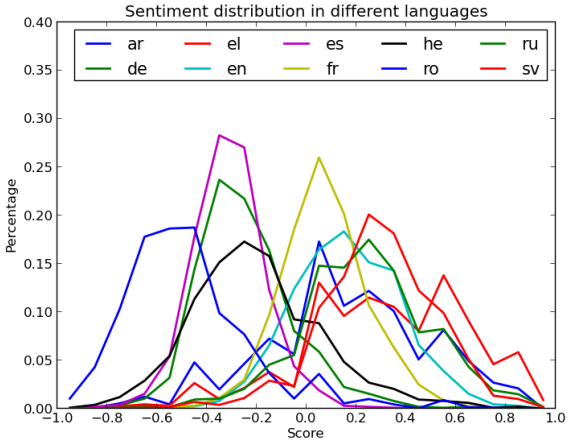
We first show that our sentiment lexicons have strong correlations in their absolute sentiment score in Figure 3.5. We pick 2 Latin languages: Spanish and French. Both Figure 3.5a and Figure 3.5b shows that sentiment scores of the same person calculated using our propagated lexicons show strong correlation.



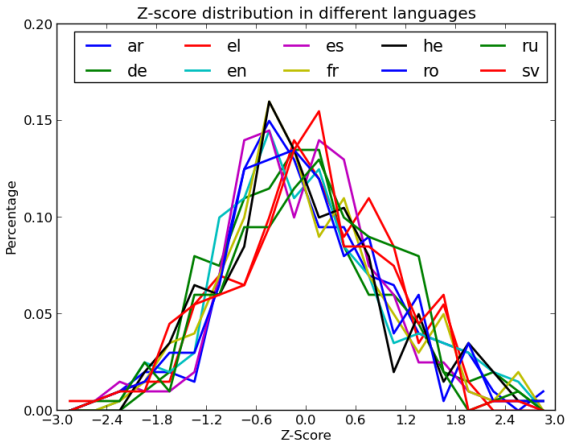
(a) English-Spanish (b) English-French.

Figure 3.5: Sentiment score distribution of Wikipedia pages between English and (left) Spanish / (right) French. We calculate sentiment scores using our propagated sentiment lexicons. The green line estimated correlations and we show that strong correlations exist in scores of the same person but different language version.

Figure 3.6 presents a distribution of entity polarity by language using our propagated sentiment lexicons. As we see that sentiment distribution of almost all languages look like a bell-shape normal distribution curve, showing that sentiment of famous people on Wikipedia fit in statistics. In "big" languages such as English or Deutsch, a famous person's Wikipedia page will always contain a paragraph of brief introduction as well as multi-paragraphs of his or her life history, which inevitably grants quite a large amount of positive words. That's the reason we see the center of sentiment distribution floating around positive side. While in languages with small group of users, people's pages might be incomplete thus only a small number of words having polarity can be detected. We demonstrates how effectively Z-score makes the distributions comparable. Combining information in Table 3.6, the differing ratio of positive and negative polarity terms means sentiment cannot be directly compared across languages. For more consistent valuation we compute the z-score of each entity against the distribution of all its language's entities.



(a) Absolute polarity.



(b) Z-score polarity.

Figure 3.6: Distribution of Wikipedia pages in 10 biggest Wikipedia languages. Top subfigure shows absolute sentiment value. Bottom subfigure mitigated differences in lexicon polarity composition by normalization, enabling the results to be compared directly across languages.

3.6.2 Consistency between language pairs

We use the Spearman correlation coefficient to measure the consistence of sentiment distribution across all entities with pages in a particular language

pair. Figure 3.7 shows the results for the 30 biggest languages on Wikipedia with first ten languages we have discussed previously. All pairs of language exhibit positive correlation (and hence generally stable and consistent sentiment), with an average correlation of 0.28.

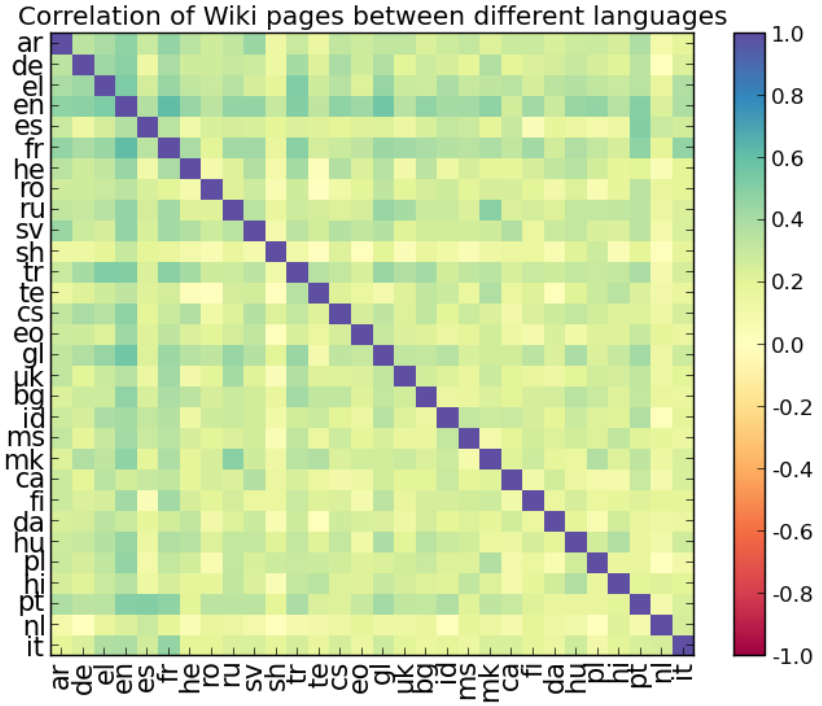


Figure 3.7: Heatmap showing sentiment correlation between 30 Wikipedia languages on 2,000 most famous historical figures according to *Who's bigger?* [88]. First 10 languages are 10 biggest languages discussed in previous sections.

We conduct consistency experiment according to Z-scores on all 136 languages. It may look confusing on some minor languages but the result seems persuasive on major languages. Table 3.7 illustrates sentiment consistency over all 136 languages (represented by blue tick marks), with the ten under discussion above granted labels. Respected artists like *Steven Spielberg* and *Leonardo da Vinci* show as consistently positive sentiment as notorious figures like *Osama bin Laden* and *Adolf Hitler* are negative. Political figures

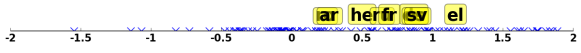
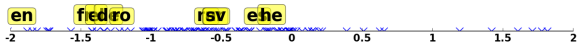
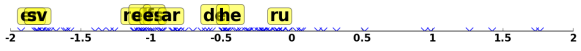
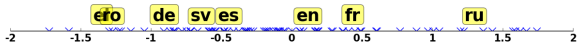
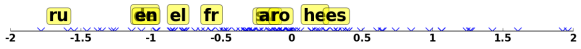
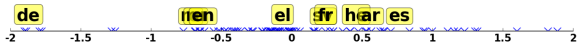
Type	Person	Z-score distribution
Good	Leonardo da Vinci	
	Steven Spielberg	
Bad	Adolf Hitler	
	Osama bin Laden	
Neutral	Hillary Clinton	
	Tony Blair	
Biased	Mao Zedong	
	Woodrow Wilson	

Table 3.7: Z-score distribution examples of typical "good guys", "bad guys", "neutral guys" and "biased historical figures". We label 10 languages with their language code and other using tick marks on the x-axis. It is obvious that our Z-score measurement keep good consistency across languages in categorizing good and bad people.

like *Tony Blair* and *Hillary Rodham Clinton* are usually neutral with some languages holding divergent opinions. There also exist biased opinions towards some historical figures like *Mao Zedong* and *Woodrow Wilson*, showing as outliners in specific languages that does not keep consistency with other languages.

3.7 Application of multilingual sentiment lexicons

We have successfully extend sentiment lexicons in English to the rest of the world. Given the fact that word-level features are usually comprehensive without interacting with grammar or sentences, we expect our sentiment lexicons to contribute consistently to multilingual world. Till now our sentiment lexicons have attracted 298 downloads with top interests of following languages: Arabic, English, Chinese, German, Spanish, Italian and French. We also received requests of minor languages like Vietnamese, Hebrew, Bangladesh and even Kazakh, showing the increasing needs of corresponding knowledge in the field of sentiment analysis for different languages.

On the other hand, we proved that propagation through Word Connection Networks is generally effective and trustful. It is possible to learn many different language features, for instance, modifiers, negation terms and various entity/sentiment attributions since similar approaches can be extended to other multilingual natural language processing tasks via Word Connection Networks with specific connections, dictionaries and seed words.

We demonstrate the consistency of our sentiment lexicons via Wikipedia sentiment analysis and such consistency makes generalized sentiment lexicon dictionary a nice weapon to handle online corpus with multiple languages combined. It would also be interesting to mine deeper for biases (e.g. Are there any differences between reputations of Mao Zedong in China and that in the rest of the World?) for same entities across regions or languages and reasoning from potential difference in cultures, histories and habitations. Our multilingual sentiment lexicons construct a playground for those ideas to be realized.

Conclusions in Chapter 2 show that word representation performs well on distinguishing and categorizing sentiments but cannot be generalized to handle multiple languages at one time. However, our graph propagation based

on Word Connection Networks shows that we can construct reliable connections of basic language features between words across languages. It could be another application to dig out more sentiment lexicons with higher accuracy and enhanced reliability considering together with word embedding, as well as training language-dependent transition-matrices to match word representations in different language spaces together and construct a generalized multilingual spaces to analyze words and their features.

Chapter 4

True Friends and False Friends: Digging Deeper into Transliterations

Transliterations play an important role in multilingual entity reference resolution because proper names increasingly travel between languages in news and social media. This process tends to create a substantial number of out of vocabulary (OOV) words in the multilingual analysis of news and social media. For instance, when "Gangnam style" topped the music charts of more than 30 countries, a word imported from Korean suddenly became part of the language spoken by millions of people around the world; news events like the catastrophic failure at nuclear power plant bring words associated with new people and places "Fukushima" across languages into common use. Plus, a large number of borrowed words have already been integrated into daily lexicons in historic culture communication, acting as bridges across languages connecting both pronunciations and semantic similarities.

Previous transliteration systems generally focus on a small number of language pairs. Further, they only consider morphological similarity even in translation systems, creating a problem of "false friends" of word pairs which look or sound alike but mean different things. We target the problem of generating transliterations between arbitrary pairs of 69 languages, and detecting borrowed words and entities across these languages. Creating such transliterations between word pairs in Word Connection Networks contributes to many language processing tasks, including entity resolution, translation, topic classification and sentiment analysis, as well as facilitates

studying linguistic phenomenon like cross-language morphologic evolution.

On the other hand, our very basic connections of exact script matching perform badly when passing sentiment messages – words with same script may not share same meaning if their pronunciations are different (e.g. "come" in English and "come" in Spanish). Grid search experiment in Chapter 3 demonstrates that we need robust connections when considering semantic transitivity, thus it is valuable to have a good transliteration system that can generalize relationship between scripts and pronunciations across languages and help constructing more reliable connections between true friends or borrowed words.

4.1 Our work

We train both transliteration models and semantic word embedding in an unsupervised manner using large-scale corpus from Wikipedia. As an example, Figure 4.1 shows our transliterations of the name *obama* into 25 non-Latin scripts. We use lower cases characters to avoid overfitting problems in leading capital letters. We provide both our transliteration (constructed from scratch) and lowest edit-distance match appearing in 100,000 most frequent Wikipedia lexicons for these languages. Our closest match proves to equal the gold standard for 20 out of 23 languages where it appears in the lexicon. Further, our constructed best differs from the gold standard name within at most one character substitution for 22 out of 25 languages.

Our major contributions are:

- **Training methods for transliteration**: We used Wikipedia to build a training set for transliteration, starting from the cross-language links between personal and place names in Wikipedia. We collect a dataset with 576,403 items contributing one or more transliterations from English to other languages yielding reasonable training and testing sets to learn transliterations. But this is a very dirty training set, because many such translation pairs are not transliterations (e.g. *Estados Unidos* for *United States*). We develop unsupervised methods to distinguish true transliterations from false, and thus clean the training set.

- **Accurate transliteration via substring matching**: We use an expectation maximization approach to use statistics of string alignments

Language	Constructed Best	Detected Best	Reference	Rank
Arabic	وباما	وبما	أوباما	7
Armenian	nբամա	օբամա	օբամա	1
Belarusian	абама	абама	абама	1
Bengali	ো৷বামা	ওবামা	ওবামা	1
Belarusian	абама	абама	абама	1
Chinese	奥巴马	大巴山	奥巴马	na
Georgian	ობამა	ობამა	ობამა	1
Greek	ομπαμα	ομπάμα	ομπάμα	1
Gujarati	ોબામા	ઓબામા	ઓબામા	1
Hebrew	ובמ	ובמה	אובמה	na
Hindi	ॉबामा	ओबामा	ओबामा	1
Japanese	オバマ	オバマ	オバマ	1
Kannada	ಾಬಾಮಾ	ಒಬಾಮಾ	ಒಬಾಮ	2
Korean	오바마	오바마	오바마	1
Macedonian	обама	обама	обама	1
Marathi	ॉबामा	ओबामा	ओबामा	1
Persian	وباما	اوباما	اوباما	1
Russian	обама	обама	обама	1
Serbian	обама	обама	обама	1
Tamil	ோபாமா	ஒபாமா	ஒபாமா	1
Telugu	ోబామా	ఒబామా	ఒబామా	1
Thai	อบามา	โอบามา	โอบามา	1
Ukrainian	обама	обама	обама	1
Urdu	وباما	ام	اوباسہ	7
Yiddish	אבאמא	אבאמא	אבאמא	1

Figure 4.1: Constructed best and detected best for word “obama” where capitalization is disabled. Constructed best is generated via cost matrices without any prior knowledge of vocabulary. Detected best is the best match in 100,000 most frequent words in Wikipedia. Last column shows the rank of gold standard reference if it appears in these 100,000 high frequency words.

to train improved cost matrices via a Bayesian probability model. Our methods employ substring matching instead of single-character transition matrices, enabling the recognition of phonemes, character bigrams, and beyond. We have trained models that permit us to construct transliterations for any string between all pairs of 69 languages. We evaluate our work against a recently-published transliteration system [31] which has been integrated into the Moses statistical machine translation system. We compare our transliteration to Moses on the four languages it supports (Arabic, Chinese, Hindi, and Russian), outperforming it in 61 of 64 standards over the set of languages.

- *Distinguishing translations from false friends*: Similarly spelled words can have substantially different meanings. Such pairs that span language boundaries are called false friends, e.g. *ropa* in Spanish means *clothes*, not *rope)*. By coupling transliteration pair analysis with semantic tests using distributed word embedding, we can generate comprehensive lexicons of true and false friends. Our methods get very good results in tests against human annotated standards for French (F1=0.890) and Spanish (F1=0.825).

  We use our approach to generate lexicons of true and false friends between English and 69 languages. We show that the lexically-closest cohort of word pairs has a higher probability of being true friends than words that are more lexically distant in 68 out of 69 languages, indicating our methods provide a good signal to identify borrowed words.

We also provide a demo that can transliterate any English string to non-Latin languages [1].

4.2 Related work

Transliteration research first associates with the field of orthographic similarity detections since pronunciation correlated with orthographic writing [18, 66, 30, 97, 56, 21]. This work shows reasonableness of character-based transliteration between close languages (i.e. languages sharing characters) but does not discuss on distant language pairs.

[1] https://soundaword.appspot.com/

Similarly, work on cognate identification also focus on close language pairs [87, 47, 86, 15, 55, 57, 85]. However, we believe multilingual transliterations contribute to even distant languages (e.g. English and Japanese) when handling OOV words and resolving ambiguities.

Further transliteration researches divide into two branches. One tries to study delicate sound changing rules of specific languages [53, 3, 92, 37, 99, 48, 43]. Especially, an excellent ideas of using Wikipedia external links is proposed in [51, 52] and achieve promising results in English-Hebrew transliteration using Moses [54]. However, all these systems are supervised and require extra linguistic background knowledge during processing. Plus, only one among this work evaluates transliteration on up to 4 languages and it is hard to generalize for multiple languages.

The other branch learns from only sequence of characters. One of the great advantages against sound based transliteration is that multilingual texts are much easier to obtain. Al-Onaizan and Knight [5] compares phonetic based systems with spelling based systems on transliterations between English and Arabic. Pouliquen et al. [81] makes transliteration model based on similar spelling rules in close languages. Recent work of Durrani et al. [31] is integrated in Moses as a module, providing an unsupervised character-based transliteration training model. Matthews [67] proposed a proper name transliteration model on several language pairs. However, we believe utilizing character-based transliteration model can provide us with even more valuable information in natural language processing tasks.

4.3 Data collection and pre-processing

Transliteration is a kind of translations with phonetically close pronunciation. It acts as a way to keep consistency in understanding a foreign word using its original pronunciation with slight difference according to sound rules of native language. Entities' names are usually transliterations, since names are hard to translate but famous people or companies need to have a reference in foreign languages. For instance, Bill Clinton was once a world-known president of United States and new lexicons will be created as references for non-English native speakers. According to a marketing evaluation [36], over 40% of the brands choose to apply transliterations when creating their brand names in aboriginal languages while only 20% of the brands choose to use meaning translations as new description so that famous cities and countries

like *Singapore*, enterprise names like *Microsoft* will usually have its unique translation or transliteration.

Kirschenbaum and Wintner [51] use titles of Wikipedia external links to build an aligned multilingual corpus. For a given language pair (i.e. English and Hebrew), they search for all Wikipedia pages that contain co-references of these languages and extract titles that shows names of the same thing in different languages. However, their work requires language-specific knowledge, for instance, they discard vowels (as a feature of Hebrew) and filter out junk data using pre-defined consonant matching between English and Hebrew.

We try to avoid such language dependent preprocessing. We query Freebase in previously mentioned categories (i.e. people, locations, countries) that are more likely to reflect transliterations in their names. In total we collect 3,388,225 entries of possible transliterations to form up a precise multilingual transliteration dictionary through Wikipedia page titles. We then perform a rough clean up procedure to (1) unify punctuation by converting hyphens, dots, comma to underscores and, (2) remove entries which do not adhere to certain formats (e.g. we accept only "first name + last name" or "whole name" for people's names). Our final collection contains 576,403 English entries with multilingual mapping.

Latin languages usually have similar pronunciations rules for common characters shared with English and there are less requirements to implement transliterations. As an additional resource, we query Google translation API to get formal translations of certain English proper nouns to all 69 languages. By doing this, we enrich resource pools in Latin languages by adopting the original orthography from Google translation and we increase the number of training examples in smaller languages where we have less Wikipedia co-references. We manually pick 1,373 entities without no multi-sense ambiguities from the names of people [19], countries and capital cities [101], resulting in more than 70,000 pairs of proper name transliteration from English. Table 4.1 shows statistics of final data size in each language. 80% of the final data will be used for training, 10% is for tuning and the remaining 10% is for testing.

Largest		Smallest	
Lang	Count	Lang	Count
French	183,270	Center-Khmer	1,585
German	178,715	Amharic	2,035
Italian	132,545	Gujarati	2,130
Polish	124,870	Maltese	2,415
Spanish	107,790	Yiddish	2,835
Russian	100,085	Kannada	3,100
Swedish	91,125	Telugu	3,840
Dutch	87,870	Swahili	4,620
Portuguese	86,515	Haitian	5,245
Norwegian	74,790	Urdu	6,115

Table 4.1: Languages with the largest and smallest set of possible entities transliterations, i.e. reflecting the availability of training data.

4.4 Training transliteration model

The purpose of our training is to get a quantified measurement of transliteration between any possible character strings in arbitrary scripts. One basic assumption here is that string pieces only match in monotonically ascending order. We expect to learn pairwise word segmentations and n-gram statistics of correlated string pieces between different languages.

We apply an EM-based method to learn transliteration rules between strings. The cost matrix is initialized so that cost of substituting any string s_1 in $Language_1$ with string s_2 in $Langauge_2$ will be 1.0 times $len(s_1)+len(s_2)$ at the beginning, including empty strings. This way each training example has a fixed cost equal to the total length of two strings.

We then start an R round EM iteration. In each round we go through all training examples and compute the minimum-cost segmentation matching according to our cost matrix. We store each observations of segmentation matching during this round in observation table, as well as matching of continuous segmentation chucks. We measure the fitness of two strings using the Bayesian setting mentioned in [89]. We consider each pair of string with transliteration relationship to be a pair of "morphemes" and we implement a simplified version without "stray morphemes" (i.e. all syllable could be reproduced in another language if allowing slight changes) and realignments (i.e. string pieces are matched in monotonically ascending order). In detail,

we calculate relative probability of s_1 matching with s_2 according to the observation table Obs. Multi-matching of similar pronunciations might reduce the value in probability thus high correlation in either direction would be considered a signal of good matching.

$$P(s_1, s_2) = 0.5 \cdot \frac{Obs(s_1, s_2)}{\sum_i Obs(s_1, i)} + \frac{Obs(s_1, s_2)}{\sum_i Obs(i, s_2)}$$

Since we include all continuous segmentations in our observation table, we can measure how good it is to "fuse" two strings together into a longer chuck for transliteration and thus decide the best split point for each string. We update the cost of matching of strings s_1 and s_2 to be the relative matching probability $P(s_1, s_2)$ multiplied by $len(s_1) + len(s_2)$. Figure 4.2 illustrates the training procedure.

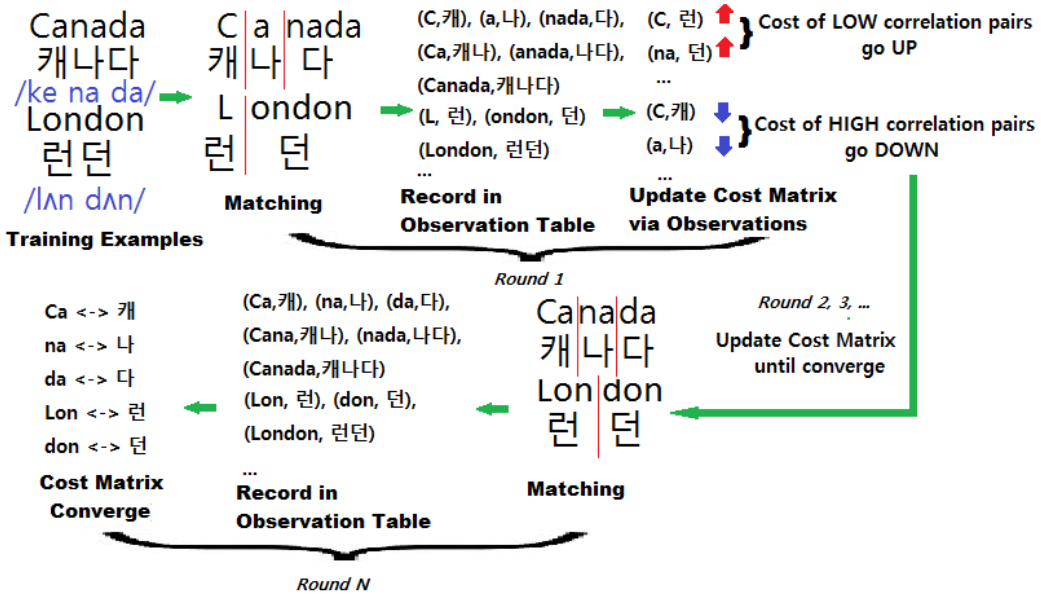


Figure 4.2: Illustration of the training procedure. Each round we compute minimum-cost-matching and record matched string pieces for all training examples and update costs matrix through a Bayesian model.

Data used in training example may be flawed as it sometimes does not reflect transliteration (e.g. *Estados Unidos* for *United States*). Such training examples act as outliners during our training and usually we cannot find any

reasonable matching even for partial string pieces. We here define "Dirtiness" to measure how many training examples are flawed in training a specific language. Table 4.2 shows 10 cleanest and 10 dirtiest languages among 69 languages. Big languages appeared in Table 4.2 (e.g. Chinese, Korean) might be "dirty", however results of these languages may not be problematic since their scales provide plenty of training examples. On the other hand, small languages with high "dirtiness" like Khmer and Amharic usually perform badly due to lack of high quality training data.

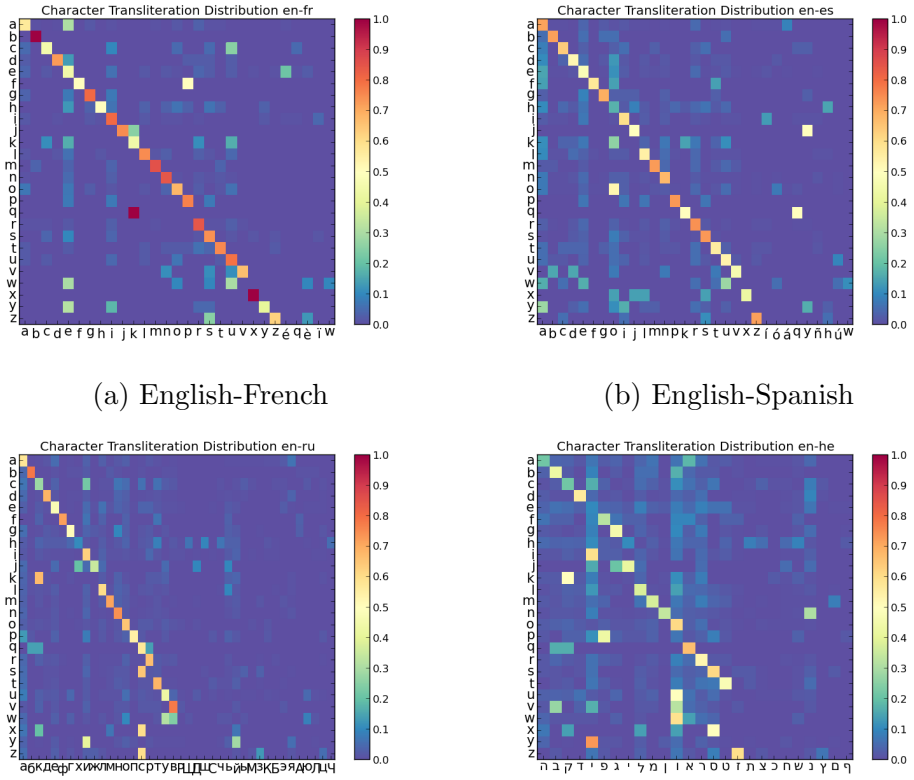
Dirtiest		Cleanest	
Lang	Dirtiness	Lang	Dirtiness
Hungarian	41.00%	Norwegian	1.23%
Amharic	36.09%	Bulgarian	1.80%
Vietnamese	32.10%	Macedonian	1.83%
Khmer	24.58%	Latvian	1.99%
Thai	24.50%	Russian	2.20%
Chinese	23.80%	Greek	2.27%
Korean	22.20%	Armenian	2.41%
Malay	20.11%	Georgian	2.60%
Tamil	18.72%	Czech	2.95%
Japanese	16.81%	Latin	3.26%

Table 4.2: 10 cleanest and dirtiest languages, defined according to the ratio of flawed examples (i.e. those cannot find correlations of transliterations) in the training set.

4.5 Glances at character matching

Here we shown in Figure 4.3 heatmaps generated from our cost matrices. Top 2 subfigures, Figure 4.3a and Fig. 4.3b, present 1-1 matching rules we discover between characters in French and Spanish. The highlighted diagonals indicate strong similarity between identical Latin characters as expected, making transliteration inside the same language family meaningless. However, these matrices reflect language differences: e.g. Spanish "y" more often acts as English "j" than English "y"; French "q" does not often match with English "q"; Spanish "b" is close to English "v" while the French "b" shows similar behaviors as English "b".

On the other hand, languages in different language families may still behave similarly. Our transliteration model discovered that Russian and Hebrew characters also have such pronunciation similarity as shown in Figure 4.3c and Fig. 4.3d, which perfectly matches with how words are pronounced in these languages.



(a) English-French

(b) English-Spanish

(c) English-Russian

(d) English-Hebrew

Figure 4.3: Probability/cost matrix for single character pairs between English and a) French, b) Spanish, c) Russian and d) Hebrew. The bright diagonal shows that we discover common equivalence for most Latin characters.

We list best matches of single English character in non-Latin languages in Figure 4.4. Some languages, like Cyrillic families, demonstrate high similarity towards English from character pronunciation and word-construction rule and we successfully recovered such information in transliteration. Other languages like Korean, though constructed in another way with different pro-

nunciations, still show reliable correlation between characters that produce pronunciations.

4.6 Experimental Results

We now show a quantified measurement of our transliteration model.

4.6.1 Baseline Model

We use the transliteration system described in [31] as baseline method to compare our results. The Moses statistical machine translation system integrates their work as a module, and allows training unsupervised transliteration for OOV words. There are slight differences between our method and theirs – we focus on generate and catenate transliteration string pieces while their method targets phrase table with context evaluation, which is finding best transliteration. We use the following parameters when configuring Moses:

- **Maximum phrase length**: 3
- **Language model n-gram order**: 3
- **Language model smoothing & Interpolation**: Automatically disabled, Interpolate
- **Alignment heuristic**: grow-diag-final
- **Reordering**: Monotone
- **Maximum distortion length**: 0
- **Model weights**:
  - Translation model: 0.2, 0.2, 0.2, 0.2, 0.2
  - Language model: 0.5
  - Distortion model: 0.0
  - Word penalty: -1

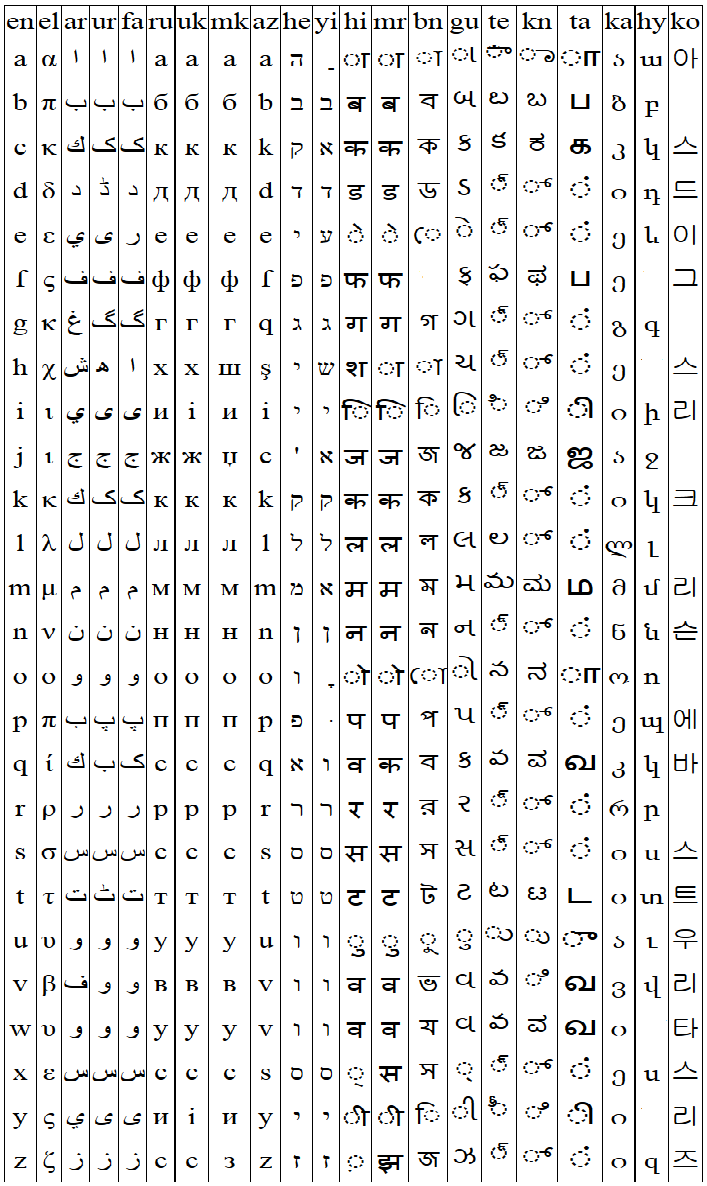


Figure 4.4: Best matches of single English character in some non-Latin languages, grouped by language families to show consistency between close languages. Languages do not always follow one on one character matching rule (e.g. Korean) but there still exist high correlation between listed character pairs.

4.6.2 Test Results

We first compare both systems trained on our Wikipedia dataset. We focus on the performance of phrase table, i.e. measurement of transliteration between string pieces since our dataset does not contain corpus context, We generate the 100-best transliterations for entries in testing set on four languages of different language families: Chinese, Arabic, Hindi and Russian.

We repeated this test using third party datasets to check consistency of training models. Here are the dataset we use:

- **Chinese**: Chinese - English Name Entity List sv1.0 (LDC2005T34) Encoding: GB-2312 Script: Simplified Chinese Number of English names: 572213 Chinese transliterations: 673385 Average number of English characters per name: 6.08 Average number of Chinese characters per name: 2.87
- **Arabic**: Combination of 10001 Arabic Names (LDC2005G02) and [20] made available for IWSLT-13. Encoding: Standard Arabic Technical Transliteration System (SATTS) Number of English names 11367 Arabic transliterations: 11367 Average number of English characters per name: 14.06 Average number of Arabic characters per name: 15.40
- **Hindi**: Indian multi-parallel corpus [80].
- **Russian**: WMT-13 data [17] and [60].

We cleaned data and retained only name mapping to feed the model, since our model does not rely on context and target a generalized method for multiple languages. Note that Moses provides several language-specific optimization methods, including weights optimizing (e.g. Mert) and Language Model Smoothing (e.g. Kneser-Ney) that might improve performance [31]. However, given our goal of unsupervised transliteration, we did not attempt to employ these in our experiments.

Figure 4.3 shows the statistics. Our system generally outperforms Moses, winning on 61 of 64 comparisons over the eight languages and metrics. The absolute closest transliteration (top-1) result only matches the translation target in roughly 1/3 of the test examples, indicating that there are typically a large number of transliterations of similar edit cost. Indeed, the absolute performance score substantially increases with top-20 and top-100 results, showing the need to reduce ambiguity through context matching. Our high

Dir	Lang	Model	Top-1		Top-20		Top-100		Levenshtein 1	
			Wiki	TP	Wiki	TP	Wiki	TP	Wiki	TP
From EN	ZH	Moses	26.7%	27.9%	44.3%	51.5%	66.1%	81.2%	64.8%	66.1%
		Ours	**30.0%**	**29.8%**	**52.4%**	**53.0%**	**85.0%**	**83.3%**	**75.0%**	**79.0%**
	AR	Moses	19.2%	20.0%	32.0%	45.0%	50.9%	80.2%	40.8%	41.6%
		Ours	**35.2%**	**25.3%**	**60.0%**	**55.2%**	**86.3%**	**83.1%**	**60.9%**	**54.6%**
	HI	Moses	23.3%	25.3%	50.4%	55.4%	70.2%	79.3%	56.1%	53.7%
		Ours	**31.5%**	**30.1%**	**61.6%**	**62.5%**	**79.4%**	**83.4%**	**61.7%**	**60.3%**
	RU	Moses	35.1%	46.1%	63.6%	69.2%	79.5%	87.5%	60.2%	70.2%
		Ours	**40.2%**	**47.2%**	**68.1%**	**67.0%**	**82.5%**	**88.5%**	**70.5%**	**72.8%**
To EN	ZH	Moses	15.6%	21.6%	32.7%	42.1%	53.5%	73.3%	45.0%	55.0%
		Ours	**24.1%**	**23.8%**	**51.1%**	**49.2%**	**80.4%**	**81.9%**	**60.2%**	**63.7%**
	AR	Moses	20.5%	20.3%	33.9%	43.9%	49.7%	79.7%	55.6%	45.6%
		Ours	**39.0%**	**26.0%**	**77.1%**	**57.1%**	**89.3%**	**82.3%**	**75.3%**	**55.3%**
	HI	Moses	21.4%	23.1%	49.8%	56.8%	71.0%	78.7%	57.3%	**62.0%**
		Ours	**29.8%**	**29.9%**	**52.5%**	**59.7%**	**79.3%**	**79.9%**	**60.7%**	60.1%
	RU	Moses	35.4%	**45.9%**	62.9%	70.8%	78.6%	**85.1%**	60.1%	70.3%
		Ours	**39.3%**	44.3%	**68.7%**	**71.8%**	**81.8%**	84.8%	**70.0%**	**70.4%**

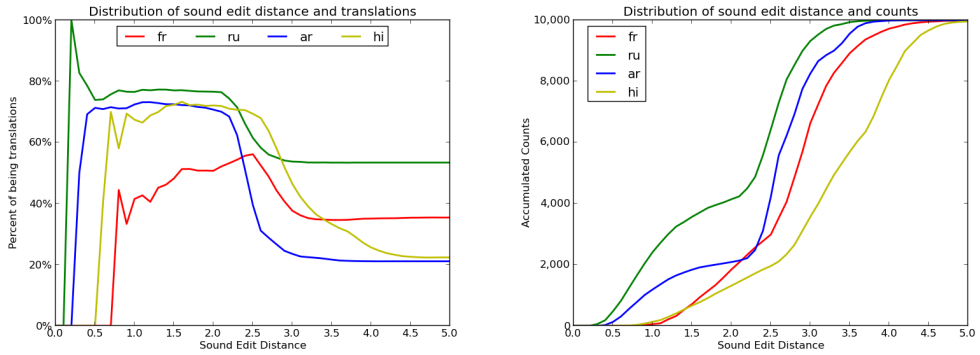
Table 4.3: Comparison of performances on Wikipedia and third party (TP) datasets. Top-k measures the percentage of correct transliterations in the top k candidates. Levenshtein 1 measures the percentage of the highest ranked transliteration that is no more than 1 substitution away from the reference transliteration, given that we consider insertion / deletion to be a special kind of substitution.

scores under Levenshtein 1 metric show that we generate reasonable transliteration for a large fraction of strings, retaining good lexical consistency with respect to the gold standard. Moses's performance substantially changes over difference training set, where we do equally well on both corpora.

4.7 True and false friends detection

We first show how we pick threshold of defining "close" word pairs. According to the way we initialize our weight function, substitution of 2 non-related characters will cost 2.0 (e.g. "mug" to "tug"). We pick threshold for each of the 69 languages with best F1-score and separate these word pairs into groups of low edit distance and high edit-distance. In order to reduce the error of machine translation, we then eliminate all word pairs with different capitalization since Google translation will sometimes treat capitalized words differently(this does not affect languages without capitalization, like Chinese).

Fig. 4.5 shows statistics of 4 languages mentioned in previous tests, including French, Russian, Arabic and Hindi. Chinese is not included since most Chinese words are less than 3 characters and are not picked in the first step. There is a drop near distance of 2.5 for all these four languages and best F1-score are picked close to this point. Since initial value of substitution between two non-related characters is set to 2, such phenomenon indicate a strong signal of "less likely to be transliteration". Threshold of 2.5 help group 20% of the word pairs into a high-similarity group while remaining word pairs are considered to be in low-similarity group, though they are much closer than random word pairs.



(a) Precision: Percentage of finding real translations

(b) Recall: Accumulative counts based on sound edit distance

Figure 4.5: Distribution of sound edit-distance from English to one other language. Figure 4.5b shows the distribution of distance measured by our transliteration system. We discovered gaps near distance = 2.0 since the initial cost of substitute one letter with an arbitrary letter is 2.0. Figure 4.5a demonstrates the percentage that close pairs judged by our transliteration system match with Google translations.

Although our transliteration model is accurate at detecting lexical similarity across languages, words that look alike or sound alike do not necessarily mean the same thing. *False friends* are word pairs across languages that look the same, but mean something different. For example, the Spanish word *ropa* means clothes, not rope. Such false friends are the bane of students learning foreign languages.

For our transliteration tests to identify true language borrowings, we

must also establish that the words have similar semantics. One way is to validate through Google translation. However, this is not the best way to handle OOV words and we decide to try if word embedding can provide supporting information. To perform such a test, we relied on the Polyglot distributed word embedding presented in [6]. The L_2 norm between two word representations captures its semantic distance.

However, the Polyglot embedding do not reside in same geometric space of latent dimensions for different languages. Thus instead of directly computing the distance between representations across languages, we check how many pairs of known translations lie within the 300 words closest words in each language in case we are lack of direct translation evidences. This process is illustrated in Figure 4.6.

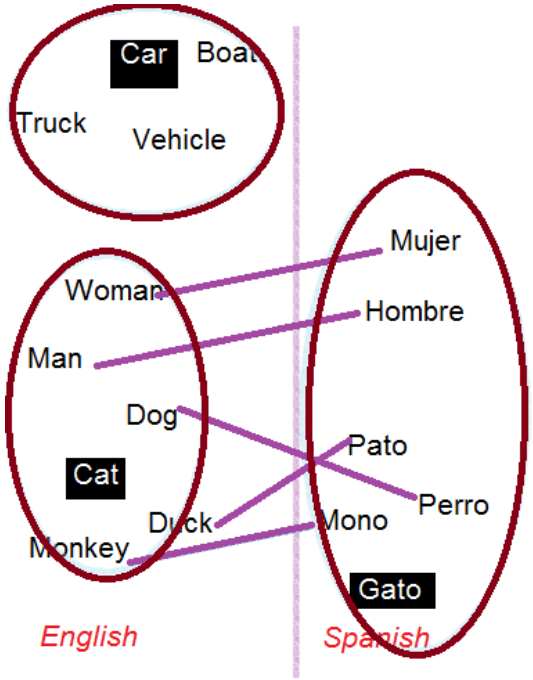


Figure 4.6: Illustration of word embedding test. In case no direct evidences of semantic similarity between "Cat" and "Gato" are found, we check number of translations that links nearest neighbors of "Cat" and "Gato" . Since ("Dog", "Monkey", "Duck") matches perfectly with ("Perro", "Mono", "Pato"), we can judge that ("Cat", "Gato") has close semantic meanings. ("Car" , "Gato") will definitely fail this test.

4.7.1 Evaluation against Human Annotation

For two languages (French and Spanish) we found published lists of true and false friends with English. We did an evaluation of our results against these human-annotated gold standards, in particular 1756 French-English cognates and 541 false friends suggested in [47, 28], including [102, 95], as well as 1345 of Spanish-English cognates [91] plus 217 false friends [32].

Our performance of true and false friends distinguisher is shown in Table 4.4. Our methods yield substantial agreement with these published standards, demonstrating the general soundness of our approach.

Lang	F1	Acc
French	0.890	89.2%
Spanish	0.825	82.3%

Table 4.4: Accuracy and F1 score for our true and false friends distinguisher using Google translation and Word representation.

4.7.2 Cross-Language Scan

Emboldened by these results, we performed a search for lexically/semantically similar words between English and all 69 of our transliterated languages. The results appear in Table 4.5, showing statistics of true-friends and false friends we detected, showing significant differences between behaviors of words with low sound edit-distances and those with high sound edit-distances:

For each language, we report the number of false friends we identify (column N). The other three columns reflect different notions of true friends: single-word translations according to Google (TP), near neighbors in embedding test (EP), and those which survive both of these semantic tests (B).

Without a language-specific analysis of each of the classes, it is difficult to determine which of these reflect language borrowings most accurately. The quality of the word embedding varies substantially by language, as does the quality of Google's translation support. Our preferred measure of quality is the ratio of word pairs which survive both tests (B) over all that having Google translations (B+TP). The 33 languages colored red and green all have a ratio of > 0.5, indicating the highest quality embedding. The red languages denote the five with the best embedding with the poorest five (in

Lang	TP	EP	B	N	Lang	TP	EP	B	N
Afrikaans	405	173	260	583	Japanese	1048	414	1112	854
Albanian	639	295	533	1237	Kannada	737	201	330	1173
Amharic	2	0	0	1	Korean	150	488	307	81
Arabic	618	571	922	913	Latvian	812	376	491	2188
Armenian	1141	405	449	1003	Lithuanian	489	542	473	505
Azerbaijani	854	166	284	1244	Macedonian	2926	422	1769	1667
Basque	352	184	187	597	Malay	226	140	263	1465
Belarusian	1561	291	765	1295	Maltese	149	214	84	88
Bengali	661	227	280	1310	Marathi	410	181	225	751
Bosnian	760	282	389	1392	Norwegian	216	414	486	401
Bulgarian	2231	674	2599	1562	Nynorsk	472	184	267	382
Catalan	**639**	**733**	**1255**	**1187**	Persian	979	284	653	1753
Center Khmer	21	194	1	41	**Polish**	**474**	**518**	**830**	**1294**
Chinese	**18**	**142**	**26**	**0**	**Portuguese**	**283**	**660**	**1150**	**567**
Croatian	**446**	**522**	**803**	**984**	**Romanian**	**388**	**618**	**1071**	**888**
Czech	**405**	**699**	**1142**	**387**	**Russian**	**854**	**1143**	**3344**	**1202**
Danish	**251**	**396**	**493**	**403**	**Serbian**	**1695**	**743**	**2022**	**1027**
Dutch	**208**	**431**	**360**	**449**	**Serbo-Croat**	**546**	**474**	**715**	**1442**
Esperanto	**522**	**410**	**638**	**750**	**Slovak**	**623**	**310**	**696**	**521**
Estonian	245	161	150	605	**Slovenian**	**518**	**296**	**552**	**632**
Finnish	**157**	**285**	**159**	**605**	**Spanish**	**530**	**788**	**1640**	**1115**
French	**486**	**949**	**2108**	**1804**	Swahili	113	96	70	458
Galician	**664**	**511**	**934**	**1343**	**Swedish**	**199**	**446**	**564**	**461**
Georgian	1591	425	661	2434	Tagalog	211	118	161	587
German	**174**	**640**	**674**	**1032**	Tamil	386	189	239	1121
Greek	1257	436	1042	1693	Latin	352	346	158	1184
Gujarati	556	156	224	412	Telugu	821	231	399	1322
Haitian	220	235	76	75	Thai	267	109	65	799
Hebrew	**670**	**466**	**864**	**1248**	**Turkish**	**263**	**350**	**534**	**1082**
Hindi	1187	323	725	1043	**Ukrainian**	**1718**	**781**	**3046**	**1361**
Hungarian	**227**	**141**	**244**	**2762**	Urdu	338	111	146	879
Icelandic	349	82	95	1156	Vietnamese	46	20	10	2976
Indonesian	**198**	**231**	**478**	**1802**	Welsh	248	81	107	992
Irish	152	155	57	837	Yiddish	626	206	156	803
Italian	**300**	**613**	**1272**	**581**					

Table 4.5: Statistics of True and False Friends between English and all 69 languages. TP denotes Google translation pairs which are not close in our word embedding. EP denotes close embedding pairs not recognized as translations by Google, while B denotes words pairs passing both semantic tests, N denotes false friends: word pairs which pass neither semantic test. Green languages are those with the highest ratio of B / TP, showing a significant correlation between our embedding and Google translation. At least 50% of Google translation pairs survives our embedding test for all 33 **Bolded** languages. By contrast, the Red languages are those where the embedding test performed poorly.

yellow) reflect languages with excessively small training data (Amharic and Khmer). Our methods have a particularly difficult time with Vietnamese,

which bases a misleading similarity to Latin languages at the character level.

We also show in Figure 4.7 some detailed true and false friends between English and Russian, based on judgement from our transliteration model (i.e. how close are the pronunciations) together with our sentiment similarity model (how close are the semantic meanings).

English	Russian	Translation	E-dist	M-score
Zanzibar	Занзибара	Zanzibar	1.19	0.44
Partizan	Партизана	Partizan	1.19	0.26
prefixes	префикс	prefix	1.14	0.34
suffixes	суффикс	suffix	1.14	0.71
pediatrics	педиатрии	Pediatrics	1.14	1.05
pretext	префект	perfect	1.20	0.00
proxy	Прокл	proclus	1.2	0.01
dancer	Данией	Denmark	1.15	0.00
since	синих	blue	1.15	0.00
pence	пения	singing	1.13	0.00

English	Russian	Translation	E-dist	M-score
single	сингле	single	0.00	0.27
London	Лондон	London	0.00	0.49
order	ордер	order	0.00	0.23
million	миллион	million	0.00	0.25
America	Америка	America	0.00	1.85
German	Герман	Herman	0.00	0.00
France	Франке	Franke	0.00	0.00
Roman	Роман	romance	0.00	0.01
personal	персонал	personnel	0.00	0.00
every	Эвери	Avery	0.00	0.00

Figure 4.7: Detailed examples of true and false friends between English and Russian. Right part lists words with low sound edit-distances. These words are easily confused according to their pronunciations. Left part shows words with slightly high edit distances, affected by small sound changes like inflections. Top part lists words that are semantically identical while lower parts shows words having different meanings. We demonstrate that there are high correlations between judgements from our transliteration model and sentiment similarity measurement and which part the words fall in.

4.7.3 Cross-Language Validation

Figure 4.8 provides a deeper assessment of our cross-language scan. For each language, we identified which words in its 100,000 word vocabulary were lexically very similar (edit distance ≤ 2, which is decided by initial value of substitution) to a word in the English vocabulary. We then considered the next closest 10,000 word pairs, which should also be enriched in real transliterations (by contrast, only 0.01% random word pairs have a translation link) – but less enriched than the initial cohort. There is a huge boost in finding

translations in low-distance group. The probability of find translation is more than doubled compared with low-similarity group, indicating our cost matrix measure sound consistency well. There are outliners however, including languages with small and dirty datasets (e.g. Amharic, Khmer), languages with short words (e.g. Chinese) and languages with lots of inflections (e.g. Italian) that increases the final cost of alignment. Indeed, Figure 4.8 (top) shows this to be true for 68 of 69 languages, denoted by points in the upper left triangle.

To establish that our embedding test accurately eliminates false friends, we pruned the lower half of each cohort according to the embedding test, i.e. retained only those words whose distance in embedding space was below the median value. Figure 4.8 (bottom) shows that this action dramatically shifts each language up and to the right. With the exception of three outliner languages (Vietnamese, Latin, and Maltese), well over 50% of our closest cohort are now true friends (translations). For somewhat more than half of the languages, the lexicographically second cohort is now rich in true friends to the 50% level.

4.8 Transliteration and Word Connection Networks

We have developed transliteration models that accurately match transliterations between 69 major languages. We successfully identify high frequency borrowed words among high frequency Wikipedia words that appear in Word Connection Networks. Further, we demonstrated that adding word embedding to provide a semantic test enables us to distinguish true borrowings from false friends. With such discovery we create a way of finding high-confidence cross language references for out of vocabulary (OOV) words with generalized resources. We have evaluated our transliterations against published gold standards when available and against intrinsic measures when such standards are not available. With full usage of our transliteration model, we can replace previous transliteration connections created by "exact script matching" and make them more robust and more reliable in semantic transitions, thus provide more precise semantic relationships between words in our Word Connection Networks.

However, there exist several directions to improve the future quality of

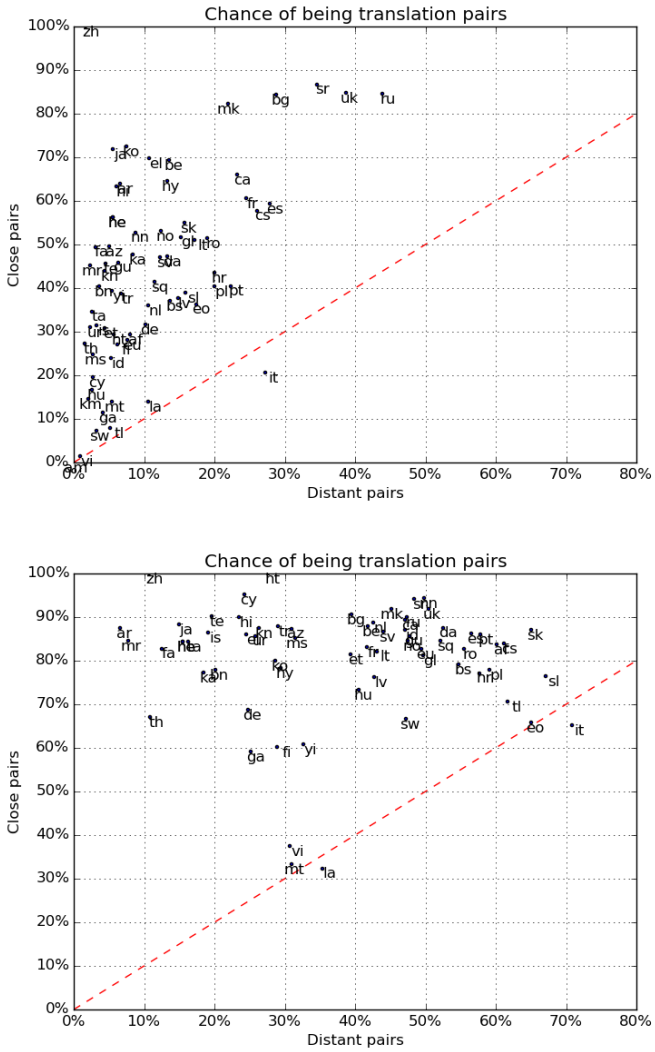


Figure 4.8: (top) Fraction of gold standard translations within very close edit distance pairs ($d < 2$) versus the next closest 10,000 pairs. (bottom). Same fractions after retaining only the 50% of pairs which are closest by embedding distance. For 68 of 69 languages, the lexically closer pairs are more likely to be translations (top). Further, eliminating pairs failing the embedding test shifts all languages to the upper right, showing that the embedding test accurately captures semantic similarity (bottom).

our transliteration model:

- **Phonetic information**: Our models improve with additional training data, particularly for resource-poor languages. An exciting way to increase this volume would be aligning speech translations as represented in a phonetic dictionary or sound system (e.g. IPA) as suggested in [48]. We did experiment on some Latin languages. The result shows that Phonetic information can improve the performance of transliteration by resolving ambiguities (i.e. increasing top-1 and top-5 results while reducing top-100 results). On the other hand, we did not collect enough resources for all other languages. Processing each different language somehow requires specific language background.

- **Multiple transliteration**: Though English Wikipedia has the richest resources in the world, it is not guaranteed that English is the source language of borrowed names. Since transliteration contains information loss, doing transliteration on transliterated words would definitely perfume badly. Currently we employ a star network of transliteration pairs centered through English. Creating another center hub using other important languages (e.g. Russian and Chinese) would improve performance.

- **Longer-range dependencies**: As we target transliteration, our model should utilize longer range dependencies to find out pronunciation rules in different languages. Observe that a silent "e" at the end of English words changes the pronunciation of vowels earlier in the word, so the "li" is different in "lit" and "lite". Our n-gram strategy somehow can figure out such features (e.g. we can distinguish "ch" in "church" and "ch" in "christmas") but that can be improved. Under context the Moses system with optimization exploits such phenomena, but we believe with we can learn such pronunciation features from the text itself.

Chapter 5

Numbers: Standing out of the Multilingual World

Numbers are totally independent from languages but play important roles in context to list fact of truth and provide supporting evidences. We might have lexicons representing specific numbers in single languages (e.g. nineteen nighty-nine vs 1999) but the most widely used numerical symbols in multilingual world remain unchanged. One good thing about numbers is that they never have multiple meanings in dictionaries. Numbers always represent the amount and they can be easily processed in any natural language processing tasks. Numbers also act as a fixed point that remain unchanged no matter in which language.

Our previous designs of Word Connection Networks do not take numbers into consideration – training procedure of word embedding basically filter out all numbers, transliteration would not work for numbers and sentiment cannot not apply on numbers. Even synonyms of numbers exist, there are no needs to adopt them. However, it is an interesting entry point to see how much information we have lost by ignoring numbers (e.g. only different digits are distinguished in training word representations [6]) and whether there are specific numbers that should not be easily neglected in natural language processing tasks.

5.1 Our work

In this work, we try to capture the historical trend of numeracy during the last 200 years which could give us extra knowledge about the history of development in real world. We studied usage of numbers in multiple time periods and we analyze preferences of precision level, including both the way people round values and the tolerance they usually have, to match historical features of events that once changed human's culture. We quantify inflation of the world from trends of numbers in multiple categories by making time series of numbers with specific units/scales and statistics of distributions of different numbers. We believe such meta-knowledge research on context of numbers could figure out implicit preferences, heuristics, and assumptions as well as knowledge context during a certain time period and thus provide prior knowledge to improve the performances of natural language processing tasks.

5.2 Related work

Related research on numeracy skills spans multiple areas in the field of social sciences, including economics, histories, cultures and education. Cohen [24] gave a brief introduction about the development of old-century-numeracy. More than two hundred years ago, precollege work emphasized standard admission requirements for universities like Latin and Greek. Even those educated men who studied in the field of higher mathematics would likely breezed through a book on basic "rules" of arithmetic in a year's time, learning formulaic algorithms for manipulating numbers in a first-level college course. Around 1800, the studies of practical arithmetic skills become far more common and Harvard University started requiring basic arithmetic for admission in 1802. Numeracy goes hand-in-hand with technological abilities and becomes a necessity for commercial economies. However, the commercial numeracy was completely context-specific that it probably retarded the development of quantitative literacy. Starting in 1812 with the intensification of market activity in the United States, an entirely new way to teach arithmetic was introduced to public which remained controversial for many decades but was proved highly beneficial for modern society.

Crayen and Baten [27] studied global trends in numeracy and its implication for long term growth during the time period from 1820 to 1949. They

evaluate people's numeracy skill using Whipple's Index, which measures the fraction of people who prefer to use multiples of 5 when reporting their own ages. Countries with relative good numeracy skills (e.g. Italy and the U.S.) demonstrate strong age-heaping characterized time series with low Whipple's Index while countries like Egypt and Turkey show a much higher Whipple's Index. They show the coefficient between the determinants of age-heaping and factors in real life, including schooling, healthiness and country GDP. They draw a conclusion that the numeracy estimates given in the Whipple's Index have considerable explanatory power: the coefficient of the Whipple's Index is consistently negative, as expected, and highly significant. Measuring numeracy can thus provide new and useful insights for researchers and add to long-term economic growth analysis. This turned out to be an interesting application of examine people's numeracy skills.

Psychology study [79] shows how numeracy skills affect decision making. They conduct experiments on groups of people with various numeracy skills, asking them to make decisions based on given information. Results show that decisions maybe influenced by merely how the outcomes are framed without any distorting. For instance, ground beef labeled as "75% lean" has no difference with that labeled as "25% fat" and "10%" is of the same chance as "4 of 40". However, people with low numeracy skills may not understand the information correctly and having wrong perceptions will prevent them from making correct decisions. Conclusions in this work support the hypothesis that adults with high numeracy skills are more likely to retrieve appropriate numerical principles and transform numbers presented in one frame to a different frame, while those with low numeracy skills will be more influenced by irrelevant affective sources.

5.3 Data collections and methodology

Google books n-grams meet our demands to do quantitative analysis on historical data. This data collection contains a large amount of number-related corpus together with historical indicators. Michel et al. [68] described the construction of Google books n-grams database from millions of digitized books. Most books being selected into the database were drawn from over 40 university libraries around the world. The text was digitized by optical character recognition (OCR) after the page was scanned with custom equipment. Over 15 million books have been digitized, which occupies 12% of all

books ever published.

Google books n-grams are usually applied to calculate the frequency of occurrences of a specific term in an area and decide the most popular words in a field during a certain period. For example, tracing back the historical data containing scientists' name, "Galileo", "Darwin" and "Einstein" may be well-known scientists but "Freud" is more deeply ingrained in our collective subconscious due to the fact that the count of "Freud" increased dramatically since 1950. Besides, statistical studies on pairs of words to get the relationship between word-pairs according to their co-occurrences. For instance, peoples' interests in "Evolution" was waning when "DNA" came along, showing a shift of topic in the area of biology in 1950s. These kinds of ideas help capturing features in old-time-corpus and open a new entry to cultural studies.

Studies using Google books n-grams provide interesting entry points of making use of statistics. Wijaya and Yeniterzi [100] tried to find the changes in the words that co-occur with a certain entity over time to analyze the semantic changes of the entity over time since such changes might not lead to identifiable surge or decline in frequency when doing statistical analysis. This kind of study would be helpful in some other applications when people need to understand the most possible meaning under a culture environment, like references correlations or micro reading, and also provide supporting evidences for historical event extraction. Kulkarni et al. [59] successfully discovered linguistic changes of words. Google books n-grams shows that the count of word "gay" dramatically increased between 1960 and 1980 which was probably caused by an entire new usage of "gay" related with several historical events. The authors contribute not only to automatically identify when changes occur but also what kind of changes occur (for instance, what topics are in transition). From their figures the most common co-occurred word-set of "gay" changed from "Young, World, Happy, Life, Lively" to "Lesbian, Men, Lesbians, Movement, Liberation" and in 1970s the new usage start to dominate the meaning of word "gay".

One issue here is that Google book n-grams split numbers with delimiters into multiple trunks (e.g. 2,147,483,647 is considered to be a 4-gram but not a unigram). Since we can only keep track of 5-grams, we record only numbers which appear in Google book n-grams with proceeding and succeeding non-number words to avoid wrong segmentations. Additionally, we need a cleaning-up to recover numbers with delimiters as unigrams to keep the whole number complete. We basically use statistics of numbers and corresponding

"units" in Google books n-grams to generate time series of numbers quoted in a specific field. Notice that "units" might appear before the number (e.g. "$100"), right after the number (e.g. "100 dollars") or after a multiplier (e.g. "100 million dollars), we process to recover numbers with multiplier as well as multi-chunk big numbers to their original numerical forms.

5.4 Time series comparison of different representation of numbers

We start our first experiment from calculating the counts of three different kinds of numbers: positive integers, positive floating numbers and text-specified integers. The counts were aggregated from unigram data and we build a time-series-line of "percentage of tokens that is a specific kind of number representation" for each kind of number mentioned above based on total aggregated counts per year. The result is shown in Figure 5.1.

The percentage of integers continuously increased since the beginning of 1800s. Originally only less than 1% of the tokens are integers and this number reaches 3% in late 1970s which shows that integer numbers gaining more importance as a dominating representation of quantity. Notice that the time series experienced a bump during early 1800s, which matches the historical promotion of quantitative numeracy in 1812 as we mentioned in the previous part. We can know from the figure that text-specified numbers appear as many as integers at the beginning of 1800s, but were gradually replaced by digit integers. Today only about 10% of the numbers are text-specified. At the same time, floating numbers gain a ratio of about 7%, while the rest 80% of numbers are digit integers.

The percentage of text numbers experienced a slow increment till 1880, and decreased slowly since then. Considering that simple numerical words are definitely used a lot when representing small numbers, as well as the word "one" has meanings other than number, we believe proportion of these text numbers would not change too much.

Floating numbers was not widely used until 1870 when the proportion of floating numbers increased dramatically. We thought the phenomenon might indicate a breakthrough that people started to have higher level of numeracy skills. Cohen [24] mentioned several specific activities that gradually extended peoples' numeracy in America: taking censuses for military and po-

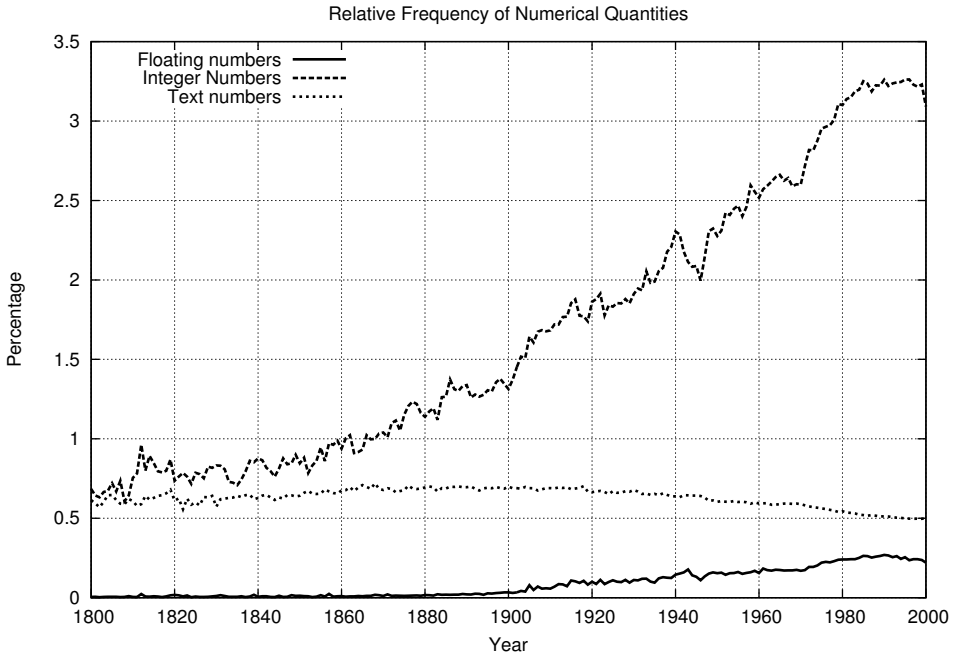


Figure 5.1: Time-series showing the trend of different representations of numbers. The category "integers numbers" contain only integers formed by digits, "floating numbers" contain real numbers formed by digits and floating point, while "text numbers" represent number in the form of English words, i.e. "nine"

litical uses, evaluating medical outcomes using simple statistics, revamping arithmetic teaching to gear it to a new commercial order, compiling numerical facts about the state("statisticks" as in descriptive statistics) to help statesmen govern, collecting voting statistics to improve the management of party politics, and finally, mounting numerical arguments in the service of the reform movements.

We got some information of text numbers from Figure 5.2. No doubt that "one" was the most important numerical word and it occupies about 0.2% of all tokens over last 200 years. The difference between the top line and the middle line shows the appearances counts of "one" per million words since 1800s. We can see that "one" appear at almost a fixed rate during the past centuries, but the fact is "one" has more meanings other than number, it might be a noise generated by other usages of "one" and we have to check the

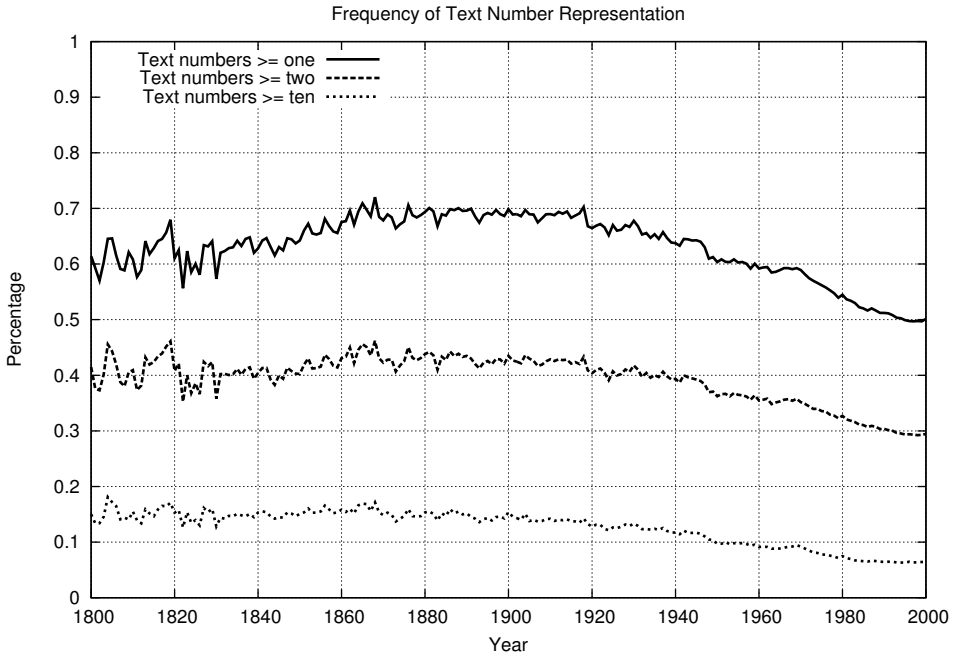


Figure 5.2: Time series of text-specified numbers, with conditions of disabling some of them. The top line is the original time series, the middle line shows the situation if we discard the word "one". The bottom line throw away all numerical words from "one" through "nine".

count of other words representing small integers. The same thing happened when we compare the middle line with the bottom line, which indicates that though the text-specified numbers are less used nowadays, there are still a lot of situations where we are willing to use them, especially for small and simple integers. What's more, one digit numbers in text format (i.e. "one" through "nine") occupy 70% of the relative usage of text-specified numbers and we guess that people are reluctant to use text since it cost much more time to record a number with 2 or more digits by letters.

There is a similar trend when considering digit integers–the shape of the time series remains same if we throw away all counts related with 1-digit-numbers. Unlike what we see when discussing text numbers, the relative percentage of 1-digit-integers was not fixed but the ratio between 1-digit-integers and total digit integers seems to be a constant around $\frac{1}{6}$. If we did not consider the usage of words "one" through "nine" with language specified

meaning, we could draw a conclusion that digitized numbers is more likely to be used when number of digits people needed to represent a number increases.

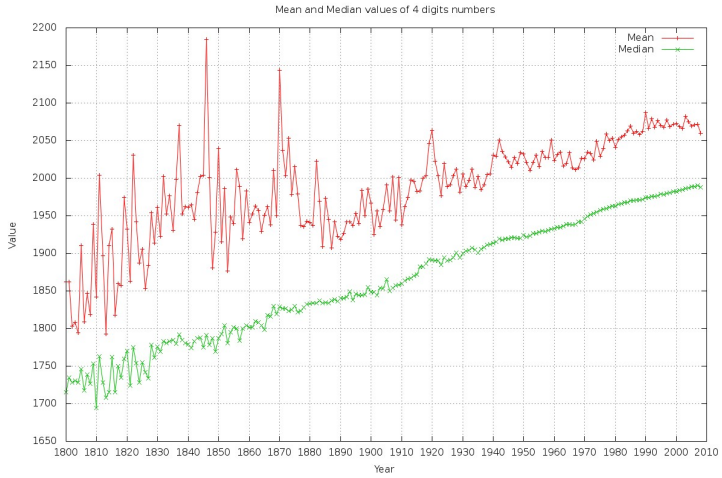
We also examine the mean and median values of numbers with different digits. According to Benford's law, 1 occurs as the leading digit about 30% of the time, while larger digits occur in that position less frequently: 9 as the first digit less than 5% of the time. We show the time series of mean and median value of 3-digit-numbers and 4-digit-numbers in Figure 5.3. The reason to pick this range of numbers is that they balance well between randomness (where 1-2 digits limit only a few possibilities) and statistics (the occurrences of 5 or more digits will be sparse). Take 3-digit-numbers for example, we expect to see a mean value close to 300ish and a median value down to 200ish in average and that remains true as shown in Figure 5.3a. However, this rule does not apply to 4-digit-numbers. We discovered in Figure 5.3b that both mean and median value of 4-digit-numbers basically increase with the same pace as current years, which support strongly the assumption that 4-digit-numbers are usually years, especially current years.

5.5 Preference of last digits

The following experiment focuses on people's preferences of last digits. Generally speaking, numbers ending with 0 or 5 are easier to remember and act as milestones in counting. We assume that people usually have a default idea of setting precisions when using numbers, especially when errors caused by differences between actual value and the rounded value can be tolerated. For instance, when talking about time, 15 minutes will probably be more common than 12 or 13 minutes even the latter one is more accurate. This phenomenon can also indicate the increasing scale of numbers people are dealing with as well as to what extent people actually care about the precision. If people do not have many chances to use numbers in large scale, they will probably round the number in order to remember it easily; otherwise they have to distinguish numbers from each other with more accurate record.

We examined all digit numbers from 1gram data and calculate count of appearances of numbers ending with 0 or 5. Figure 5.4 contains the result showing time series of people's preferences on numbers separated by different number of digits. Though the number of last digit in integer should be uniformly distributed, we can still believe that people would prefer those

(a) 3 digit numbers



(b) 4 digit numbers

Figure 5.3: Time series of mean and median values of (top) 3-digit-numbers and (bottom) 4-digit-numbers. The distribution of 3-digit-numbers basically obey Benford's law, supporting the fact that people does not have specific usage of 3-digit-numbers in real-life. However, the usage of 4-digit-numbers usually correlate with years, especially current years.

numbers ended with 0 and 5, i.e. "10th anniversary", "about 5 years ago". After checking the time series of 2-digit-numbers, 3-digit-numbers and 4-digit-numbers, we see almost fixed proportion of preferences on numbers

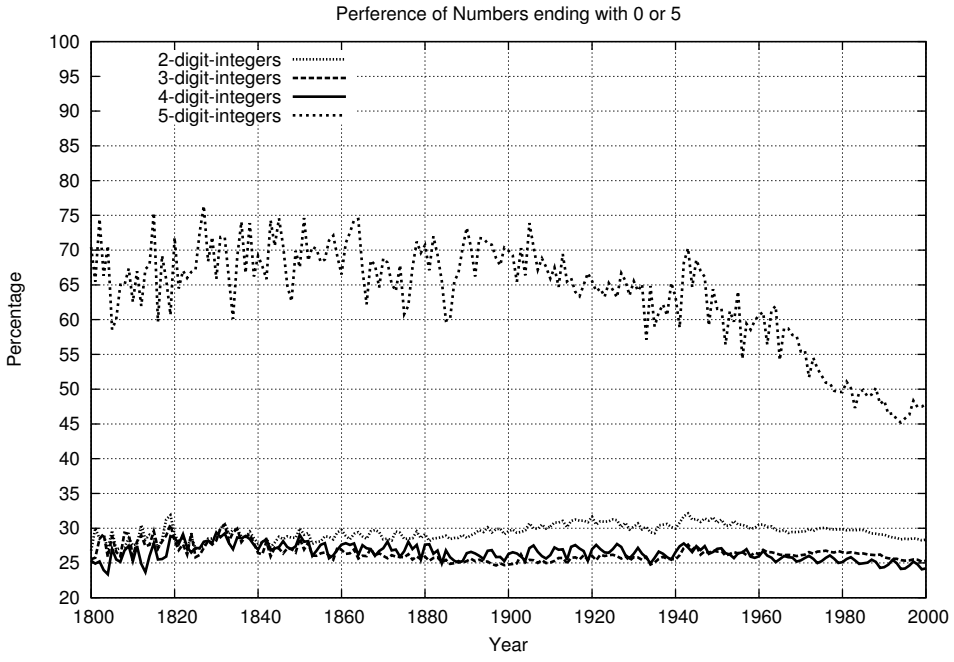


Figure 5.4: Percentage of numbers ending with 0 or 5 in different categories. Even of small numbers (i.e. less than 3 digits), there are about 30% of them ending with 0 or 5, showing a significant difference on random baseline which is 20%. 3-digit-numbers and 4-digit-numbers get closer to average due to the need of precision (i.e. year).

ending with 0 and 5 (slightly higher than expected 20%) and the preference is even higher when considering 2-digit-numbers. We kind of believe that people meet too many 2-digit-numbers in their daily life and sometimes they just pick those ends with 0 and 5 as meaningful samples. When 3-digit-numbers and 4-digit-numbers come into consideration, people can hardly figure out the accurate value. On the other hand, even if people could show the accurate value, they might be unwilling to do that since everyone can tolerate the error.

Interesting things happen when considering 5-digit-numbers. Over 50% of the 5-digit-numbers end with 0 or 5, which means people did not even care about the accuracy under most conditions when using 5-digit-numbers. We checked and counted the appearance of all 5-digit-numbers in our raw data and found that the following four have heavy weights: 10000, 20000,

30000 and 50000. No doubt that these could not be accurate expression of value in real life, and we can only explain that these numbers are used, most probably, as a rounded or estimated value that help people understand the scale clearly. Even if the line shows that there is a trend of making 5-digit-numbers more accurate recently, which implies that people are dealing with more and more large numbers and the distribution of last digit is becoming more uniformly shaped, 5-digit-numbers are not so common in people's daily life as an accurate count. Also, the phenomenon gives us a hint that people only have a direct accurate idea of numbers on a scale of thousand. That may reflect the reason why people choose to separate big numbers by comma every 3 digits in English.

5.6 Quantities with units

The next several experiments deal with numbers that appear together with a defined unit. In order to finish this task, we firstly tried to extract all specific types of 5-grams, whose last token represents a common unit in some categories and tokens preceded the unit indicate a value. Secondly we convert all different units in the same field into a fixed and most common one for the sake of easy comparison.

5.6.1 Discussion of weight

For instances, in the statistics of mass value in people's daily life, we'll convert all SI units into grams and all English units into pounds–we kind of believe that these are the most appropriate units according to people's daily experiences, even that the basic SI units of mass is kilograms. In the following graphs, X-axis will show the logarithm scale based on chosen unit. Y-axis shows the accumulated percentage of total counts starting from the minimum size people might talk about. The total time period between 1800 and 2000 will be separated into 5 sub-periods of 40 years in order to show the features appeared in a specific time range. We can easily imagine that majority of the data will fall within a small range since over 80% of the discussion in a field will talk about a predictable unit scale, we may sometimes put our interests on the larger-side-tail and the smaller-side-tail–the two parts that reflect certain culture changing but not considered to be majority. These parts will show differences between different time periods.

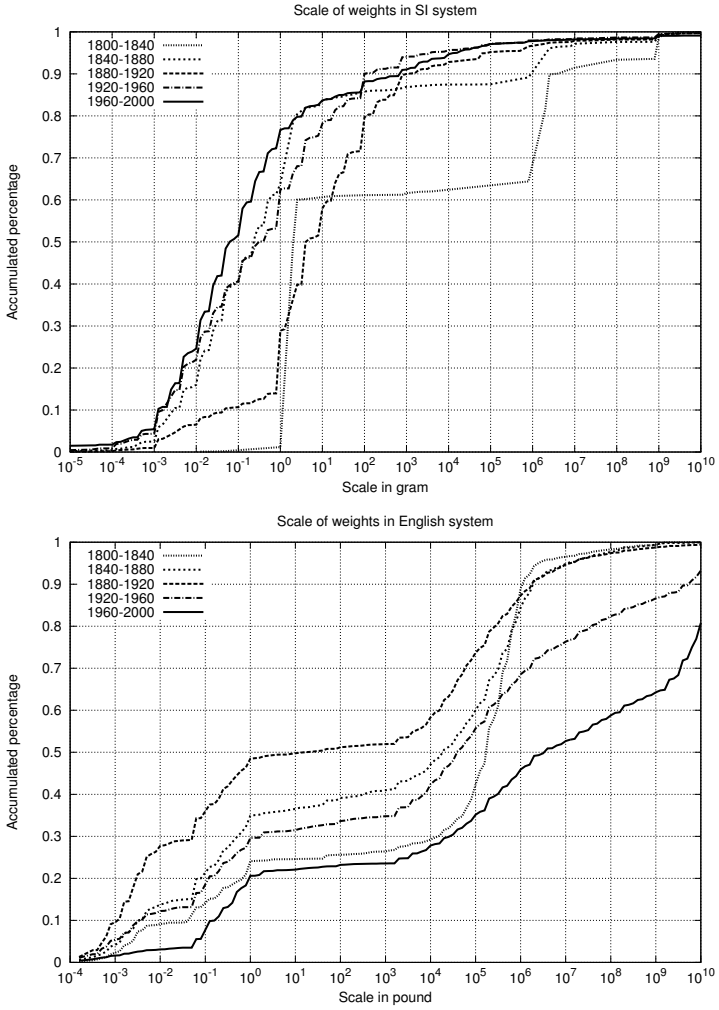


Figure 5.5: Accumulated distribution of weights in (top) SI mass system, unit gram and (bottom) English mass system, unit pound. The trend of accumulation seems similar but the gap position indicates that there exist huge differences in how people use these two systems in daily life.

In Figure 5.5 (top), the right part shows the larger-side-tail of the mass that people are talking about in different time periods. Less than 5% of the total mass discussion are focused on things that is heavier than 10^4 grams, which is about 10 kilograms. Comparing the latest time line (1960-2000) with ancient ones we see more discussion on heavier things. The left part

shows smaller-side-tail of the same graph, implying that among 5% of the time things less than 10^{-3} grams (1 milligram) was talked. The line shows when the time come approach, people have more interests in things whose weights are not perceivable previously. All lines show the same trend that majority of people talk about things starting from the level of 0.01 gram. The 99th percent line of weights increase from 10^9 grams (1,000 tonne) to 10^{18} grams, which shows people come up with a totally different world during the last century. It also shows that the count people use "gram", "milligram" and other smaller units in SI system is significantly greater than the count people use "kilogram" and "tonne"–that's why the count of heavier weight is not small but the proportion is still low.

Similar things happen when we consider the English system of mass (pounds, ounces, grains, etc.). Since people seldom use English system to measure light things recently and the unit "grain" is rare, English system care less about things that can be measured by "pound". We can see that since people care less about light weights in English system, the proportion of heavy item is higher than the one in SI system. The heaviest thing is about 10^{15} pound, stay almost on the same level of SI system. In Figure 5.5 (bottom), we found that shapes of lines did not change much for all 5 different time periods except the smaller-side-tail and the large-side-tail, showing that things people care about in real books were similar during the past 200 years.

5.6.2 Discussion of lengths

We conducted similar experiments on "length" field using both SI system and English system. The results are shown in Figure 5.6. On the larger-side, since the English unit "mile" is still used widely, there should be no significant difference between the 95% percentage number, which is about 10^6 feet (around 300 kilometers), but it seems that English system do not have popular micro units that match "micrometers" and "millimeters" (once there was "micron" but less popular now), thus the 95% point in SI system is smaller, only reach 10^5 meters (100 kilometers). But the longest thing mentioned remained on the same level in two different systems, both increased to the level of 10^{14} meters. On the smaller side, the lines show exactly the same trend as time. Recently people care more about smaller size. And 10^0 feet start to become a threshold of popular discussion. When considering SI system of length (meters, kilometers, millimeters), since it again focuses on

delicate units and we can see large counts related with this scale and it shows almost the same trend as the experiment result on mass.

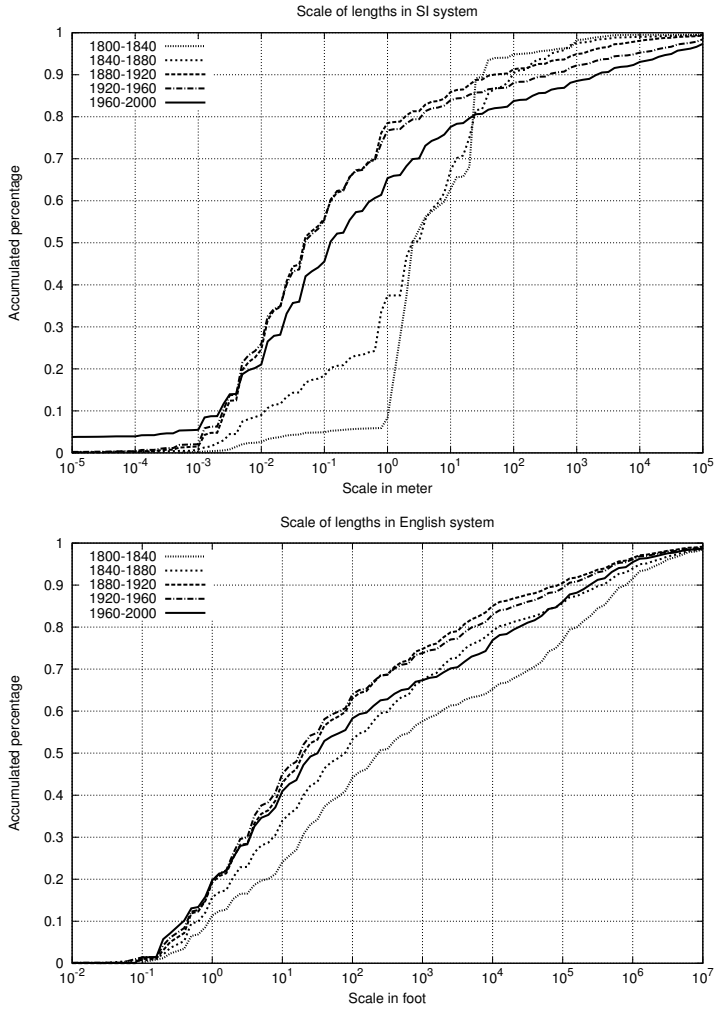


Figure 5.6: Accumulated distribution of lengths in (top) SI length system, unit meter and (bottom) English length system, unit foot. We see similar distributions and changing points as the weight system.

What's more, we did an additional experiment comparing all length data in a certain range (from 10^0 to 10^5 meters) and see whether choose to user different system of units will share the same accumulated percentage style.

From the results in Figure 5.7, we can find that in the last 100 years,

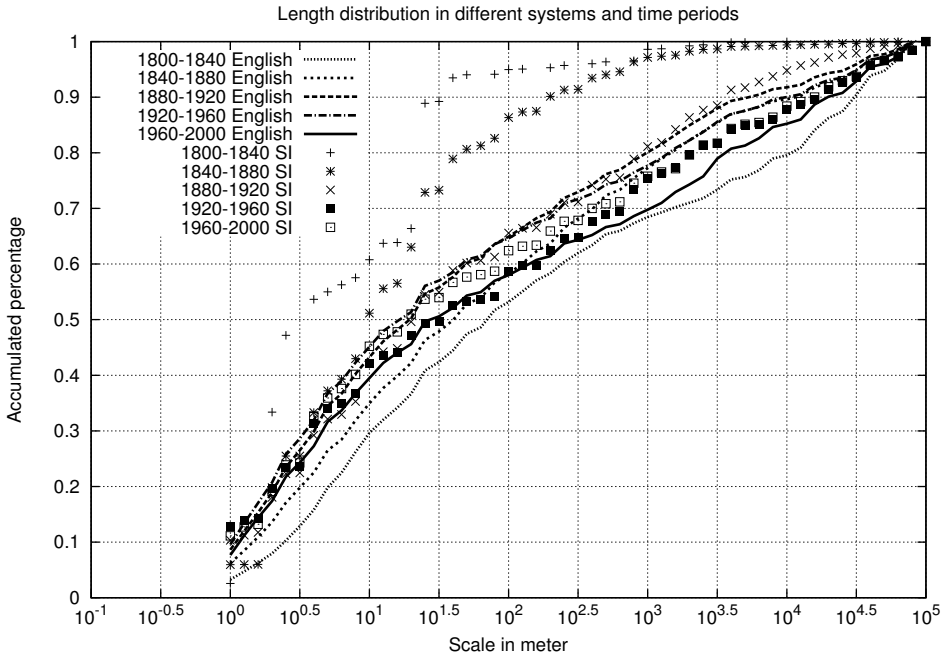


Figure 5.7: Compare distributions of object lengths between different length systems. Dots show lengths distribution using SI units within the range of 1 meter to 100 kilometers. Lines show the lengths distribution within the same range but use corresponding English units.

there are almost no differences in items length distributions between English system and SI unit system. Such prefect matches suggest that nowadays, even using different measuring units, the related objects are almost the same. But in the old age, for instance, the beginning of 1800, people seem to have their own point of view on measuring. One possible reason is that people are not that agree with using floating numbers in daily life–which causes "gaps" between consecutive values. In order to accurately measure length of an object, there might be a preference of using different systems for different range of lengths.

5.6.3 Discussion of currency

We also tried the experiment on money quantities dollars as shown in Figure 5.8. In this experiment we see a totally different shape, indicating a fast-

developing world of measurement. From the figure we can see that the 95% line was $10^{8.5}$ dollars (316 million dollars) 40 years ago, while the most recent data has the line of $10^{9.5}$ (3.16 billion dollars). In old ages people never talk about such big quantity of currency, which shows that people are not only dealing with more and more large quantities in currency, but also suffering from severe trend of inflations. If we consider that the inflation rate to be a constant, then the estimated inflation rate would be 106% each year. On the smaller side, people do talk about money starting from cents, but only in a very small portion. Additionally, once in history there exist currency unit of "half cent", but we found no such evidences in our Figure.

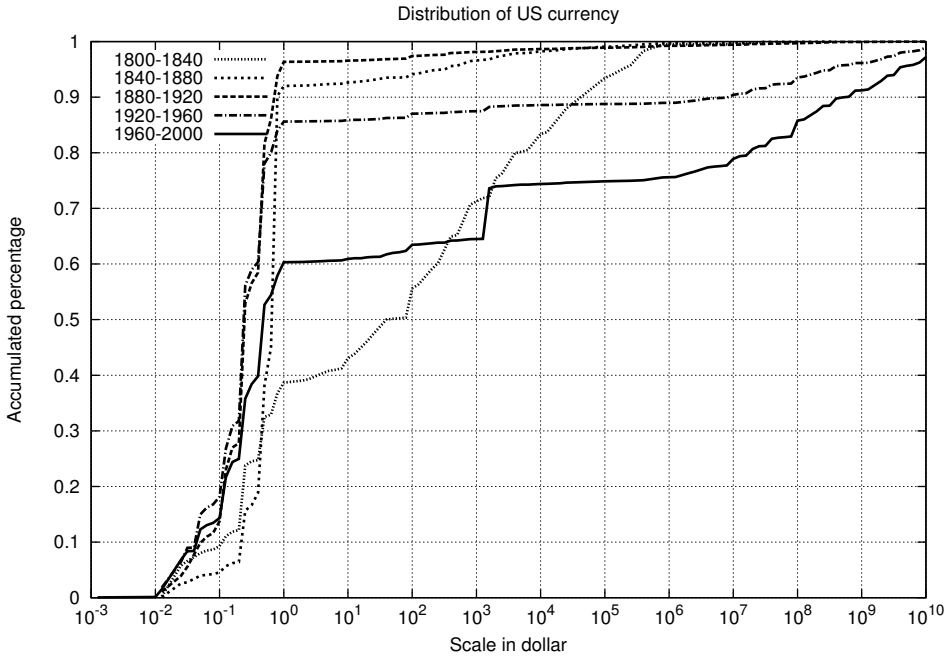


Figure 5.8: Accumulated distribution of US currency, unit dollar. The shape is totally different from what we see in weight systems and length systems as human world of currency changed a lot in past 100 years.

5.6.4 Discussion of time

Finally we have the result in time field. No doubt that we can trace back our history to the beginning of the universe, but when people talk about time,

we usually discuss on the level of minute as shown in the experiment result Figure 5.9. We also see the trend that people are getting more interested in longer time periods, especially those longer than human histories – longer time periods are mentioned more during the development of science and technologies.

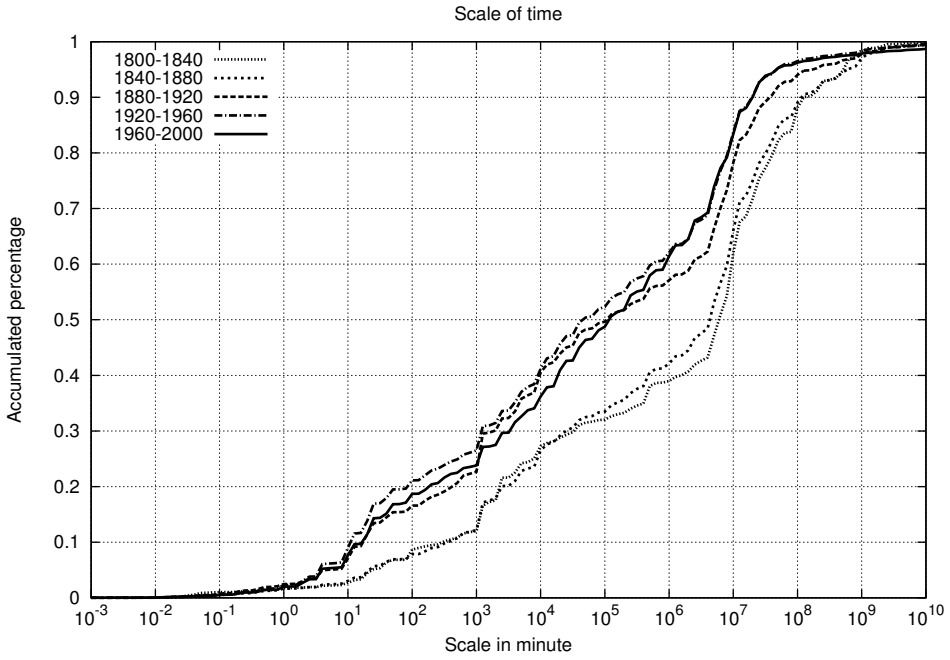


Figure 5.9: Accumulated distribution of time, unit minute. Longer time periods are mentioned more during the development of science and technologies.

Another experiment on quantities with units studies the value before a certain unit only, to see how people thought about number usages before units. We examine the distribution of numbers before a certain weight unit gram and the observation is, if there is no larger units in the same field, people tend to add "thousand" or "millions" before the largest unit in order to represent a even larger scale; otherwise, people prefer to use a value ranges in 10 times the level of an adjacent units. For instance, if people would discuss a time period of 150 minutes, he would rather say 150 minutes or 2.5 hours; if the time period is 730 minutes, people would probably change the unit to "hours" and round it to a certain extent, which is 12 hours, instead. If in one of the expression the value is perfectly rounded like "800 minutes" or "25

hours", such kind of expression will also be preferred. These phenomenon might explain the appearance of "gaps" when consider only several possible units since people might use another expression for better memory. With an appropriate unit, people could easily have an idea of the scale and thus understand the meaning better.

5.6.5 Quantities without defined unit systems

There are also some interesting nouns like "people", "cars", "towns" that can imply the development of human society. The following Figure 5.10 shows the number scales and the distribution of values of word "people". The smaller tail start from 1 (i.e. 10^0) –though there might be some irregular or special expression of related with people since talking about 0.12 people is meaningless.

No doubt that the total number of human beings on Earth is rapidly increasing, but more than 1% of the information talking about people involves a large scale (10,000 or more). As early as 1800s, the line of 95% reaches 1,000 people, which shows that the quantitative numeracy developed in the old age for many practical reasons. It also indicates that people need more sophisticated numeracy skills to handle with this information even in the early 1800s to do census correctly, as well as other government issues.

We also examine the most popular words appeared after a number and we show our results in Table 5.1. According to our previous experiment, we put these words into three different categories: meaningful units, meaningful non-unit-nouns and words without meanings. Here meaningful units specifies a unit in a known category like "kilogram" or "feet", units represent something that can be described and listed like "pages" and "volumes". Note that we put the words like "millions" and "times" into this category. We thought that it is correct if we only have three categories, but there is another idea that create a new category for such words because they are probably used to describe the noun following them–such examination need more knowledge about 5grams since phrases like "several millions of ADJ NOUNS" would appear quite often and occupy each possible position of token–and we now simply assume that the appearance of non-unit-nouns and units remain the same proportion as if we do not have these extra modification words. And finally we have the third category, which contains words like "the", "a", "of", "in". These are probably generated by the meaningless fragments of n-grams – since the raw data did not consider any situation that two adjacent

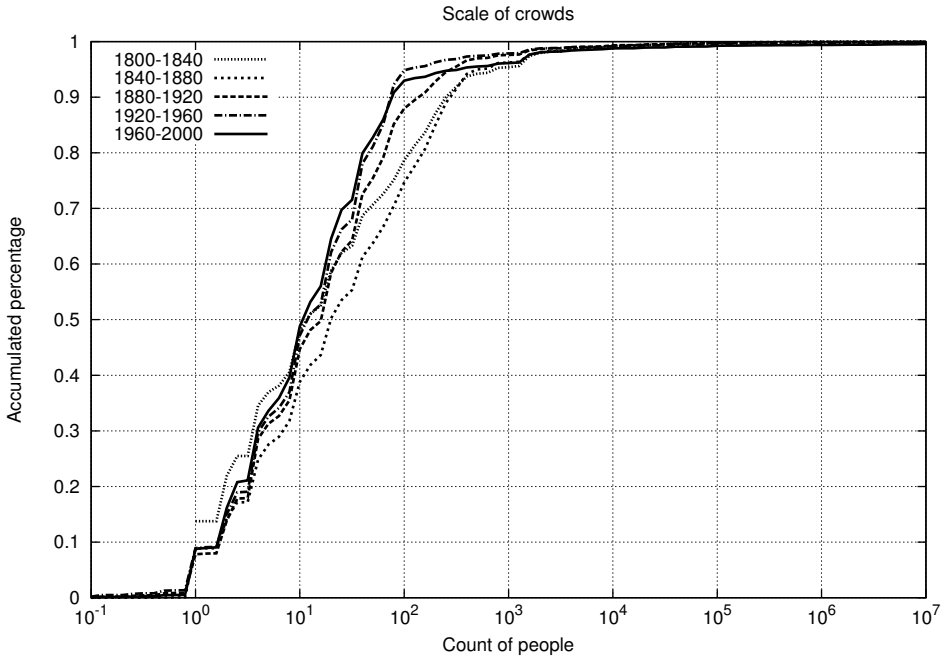


Figure 5.10: Distribution of numbers appearing before the word "people". There are some outliners talking about meaningless fraction number of people, however majority of values ranges from $10^0 = 1$ as expected.

words belonging to different short sentences, i.e., "In 1940, he started his new life...", we simply use them as references.

From our results we know that the proportion of reference sentences are generally increasing and we thought the reason was that numbers are given more usages other than acting as a number before a noun in recent decades. Worlds are more digitized and even a number itself can become a source of information, for example, the final score of a basketball match. Besides, the number of unit-abbreviations increased dramatically in the last century– statistics shows that 5 out of 10 most popular units are abbreviations, supporting that people are dealing with a wider range of data from another aspect. People should have a more systematic and efficient way of understanding these units. What's more, we can make a conclusion that large numbers and statistical usage of numbers appear more frequently in last 40 years–not only due to the increasing number of appearances of word "billion" and "percentage", but also due to the bump of words that represent a

month, like "November" and "April". No doubt that the number followed by a month will represent a date. If we take the fact that there are 4 units related with time appeared in the most popular units list in to consideration, we can make a conclusion that people now have a powerful database that can trace back in history and trying to make their decision based on these statistical information.

YEAR	Most Popular Unit Words	Most Popular Nouns	Proportion (Refs/Units/Nouns)
1800-1839	feet, miles, years, inches, acres, dollars, degrees, days, tons, pounds	men, vols, chapter, inhabitants, chap, parts, persons, guns, millions, times	63.82%/23.76%/12.43%
1840-1879	feet, miles, years, inches, tons, cents, acres, ft, days, m	vols, men, chapter, parts, vol, pages, inhabitants, persons, times, kings	64.12%/22.88%/13.01%
1880-1919	feet, miles, inches, years, cc, ft, cents, pounds, tons, mm	vols, men, chapter, illustrations, pages, parts, cases, shows, times, persons	69.33%/21.89%/8.78%
1920-1959	feet, years, miles, ft, mm, days, inches, hours, m ,ml	million, vols, shows, men, times, cases, chapter, pages, american, persons	77.21%/16.70%/6.09%
1960-1999	years, m, mm, mg, hours, cm, ml, minutes, months, miles	million, shows, patients, figure, men, fig, cases, people, chapter, billions	75.59%/17.26%/7.15%

Table 5.1: Statistics of most popular words after numbers in different time periods in past 200 years.

5.6.6 Benford's law

Benford's law, also named as first digit law, states that in lists of numbers from many real-life sources of data, the leading digit is distributed in a specific, non-uniform way. According to this law, numbers with first digit of 1 appear about 30% of the time. Larger digits occur as the leading digit with lower and lower frequency, to the point where 9 as a first digit occurs less than 5% of the time. From our statistics, the median value among all integers with certain number of digits always stay between the number start with 2 and the number start with 3, which match with Benford's law that numbers starting with 1 and 2 occupy around (30% + 16% =)48% of the time.

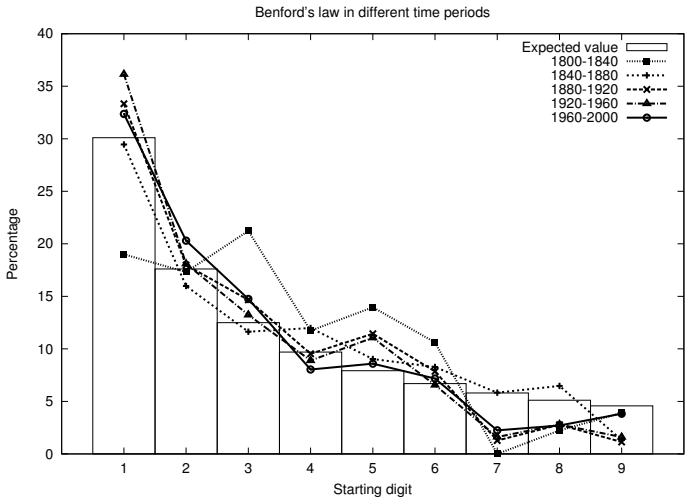


Figure 5.11: Expectation and observation of numbers starting with different leading digits that connect to unit "meter", grouped by five 40-year-periods. We find only minor gaps between expectations and observations except the earlist period of 1800-1840.

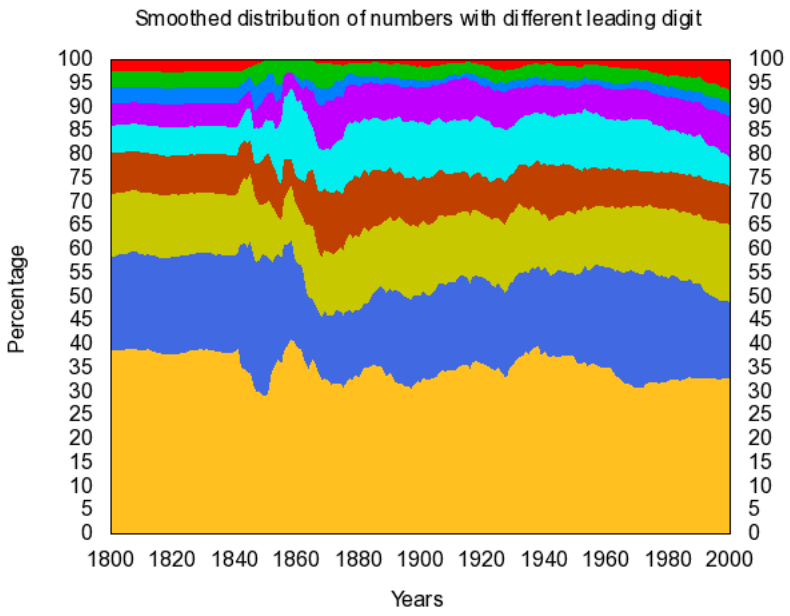


Figure 5.12: A stacked area graph showing the verification of Benford's law. Each strip represents the percentage of numbers starting with a certain digit, in the order of 1, 2, 3, ... , 9 from bottom to top.

Figure 5.11 shows the statistics of all numbers followed by the unit "meter" according to their first digit number in scientific notations. Since "meter" might be the most common length unit in human being's daily life, we can simply assume that these numbers cover a quite big range of common objects.

According to Benford's law, the probability that a number start with d should be about $\log_{10}(1 + \frac{1}{d})$, thus the probability of a number whose leading digit is 1 would be around 30%. And we can match the result number with the width of the band in the graph to basically verify Benford's law–it seems quite correct based on the previous two figures. We also found that in Figure 5.11 the observed counts of digit 7 and 8 seem less than expected. The reason would probably be "gaps" of possible value, given the fact that 7 meters or 70 meters might be less popular in real world (too short for distance measuring, too long for object length, and even not good for rounding). We thought it to be tolerable error if considering the habit of manually rounding, which was similar as one of the situation mentioned in [74]. Figure 5.12 shows time-

series using all values in the field of length without converting them into a fixed unit, i.e. distinguish "2 miles" from "3.22 kilometers". We expect to see the value (number only) distribution with multiple different units instead of a single "meter". This time the proportion of digit 1 was little bit higher, showing that people prefer memorize or record numbers with seemingly small starting digit even if the actual values are exactly the same.

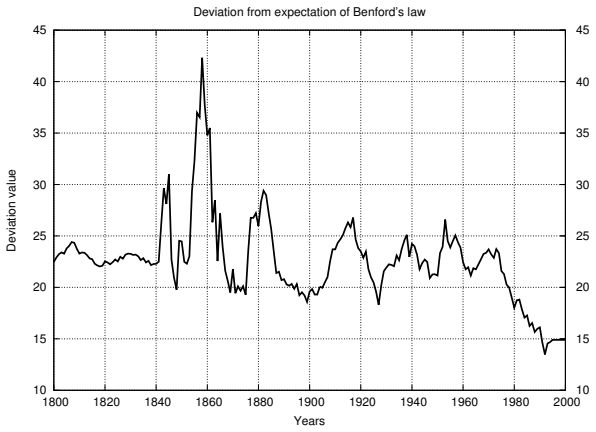


Figure 5.13: Deviation from expectation over time indicating when the usage of numbers suddenly changes. Peaks usually reflects the process of replacing old arithmetic habit or increasing the digits used for counting due to development of businesses.

We use a simple method to calculate the deviation of actual first-digit-distribution from expectation of Benford's law on all length values. The formula we use was:

$$D = \sum_{k=1}^{9} |O_k - E_k|$$

where O_k represents the observed percentage of numbers starting with digit k, and E_k shows the expected percentage by Benford's law.

In Figure 5.13 we see quite a lot of noises during the period between 1840 and 1890, which we thought reflects the process of replacing old arithmetic habit. The global trend is that the deviation get smaller and smaller, showing that we have more data, more reliable measurement and more reasonable distribution of numbers.

Finally Figure 5.14 shows smoothed distribution of number usages. The time-series seem to be nicely smoothed and matched well with the expec-

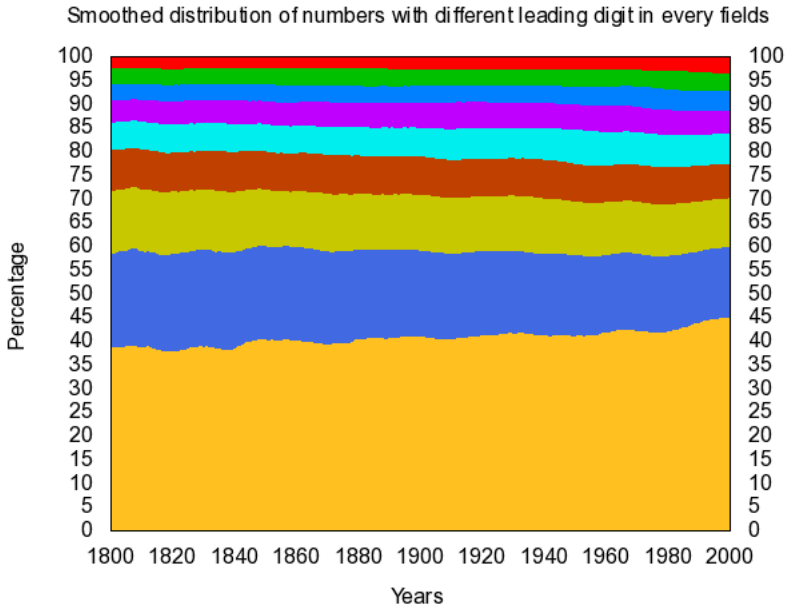


Figure 5.14: A stacked area graph showing the distribution of all numbers regardless of the units. Layers from bottom to top represent 1, 2, 3, ... , 9. We see clearly from the graph that smoothed distributions of numbers match pretty well with Benford's law, indicsating that most of the numbers are used for naturally developing businesses.

tation of Benford's law. However, the proportion of digit 1 is even higher. After carefully examine the data, we found that values in the field of time dominate the counts–we have a lot of 4-digit-number representing year information. Plus, we have a special 60 base representation describing minutes and another 12 base (or 24 base) representation for hours where the original Benford's law does not work well. All these provide us with many values starting with 1 and reduce the appearances of 7, 8 and 9. The dominating counts also eliminate possible fluctuation and it could hardly be changed as a fixed cultural behavior.

Chapter 6

Comparing Historical Figures using Wikipedia

People who have their names on Wikipedia are usually historical figures with impressive contributions in certain fields. Effective analogies among these famous people arise continuous interests of questions like "Who are the next Lincolns, Einsteins, Hitlers, and Mozarts?". However, analogies of historical figures usually combine multiple facets of these individuals, including shared personality traits, historical eras and domains of accomplishment. Figure 6.1 gives the closest analogies examples on different aspects of *Isaac Newton*.

Analogies are of course highly subjective, and hence rest at least partially in the eyes of the beholder: "there are a thousand Hamlets in a thousand people's eyes". We are interested in building a generalized model to find all these candidates with high similarity level based on connections and semantics in their Wikipedia text. We would also like to automatically rank categories tags – the human annotation that make best summarization – in their Wikipedia text to give an clearer idea of why these people are memorized. It could be very evocative when correctly identified examples like: *Martin Luther King* and *Nelson Mandela*; *George Washington* and *Mao Zedong*; *Babe Ruth* and *Sachin Tendulkar*.

In this work, we propose a method for identifying historical analogies through the large-scale analysis of Wikipedia pages, as well as ranking human-annotated Wikipedia categories by their impressive level using combined resources of raw text and Wikipedia internal links. Over 600,000 historical figures have associated Wikipedia pages in English alone, a population greater than San Francisco. Our methods readily generalize analysis of the

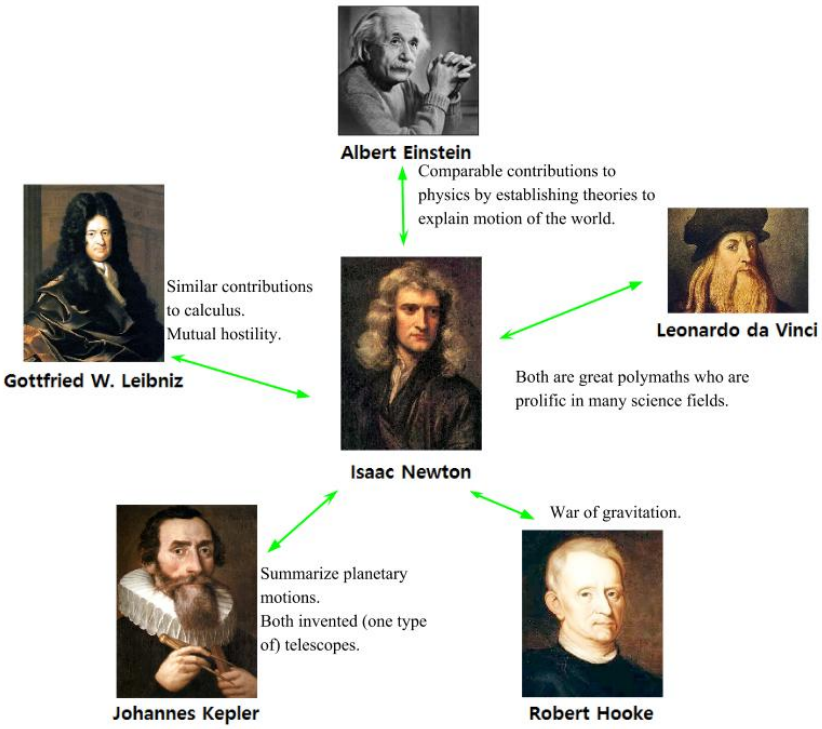


Figure 6.1: Sample analogous historical figures of Isaac Newton and corresponding explanations of similarity. Analogies are highly subjective thus it is impossible to find perfectly fair and objective gold standards.

Wikipedia available in over one hundred other languages as well.

We first develop a baseline reference standard for historical analogies to judge the effectiveness of our methods, which estimate "similarity between people" by the number of shared Wikipedia categories. Later on we ask volunteers to vote for the most important categories and use such data to calibrate importance level of each category. There are few contradictions between these two criteria. However, adopting importance of categories makes our evaluation more consistent with human beings, thus providing an even more reasonable definition of 'similarity" and better quantify performance of our similarity detection algorithms.

The most obvious applications of this are in historical interpretation and education, but we believe that the problem runs considerably deeper. *Ho-*

mophily is the tendency of individuals to associate and bond with similar others. Being able to identify similar individuals thus goes to the heart of algorithms for suggesting friends in social networks, or even matching algorithms pairing up roommates or those seeking romantic partners.

Specifically, our work makes the following contributions:

- We investigate four different unsupervised approaches to representing the semantic associations of individuals: (1) Individual word frequency using TF-IDF, (2) Weighted average of distributed word embedding, (3) Topic analysis using LDA (Latent Dirichlet Allocation) and (4) Deepwalk embedding generated from Wikipedia page links. All proved effective, but Deepwalk embedding of Wikipedia links yielded an overall accuracy of 91.4% in our evaluation. We create vector based representation of about 557,965 people on Wikipedia and measure their similarity. We provide an interactive demonstration of our historical analogies at *http://peoplesimilarity.appspot.com/*, where you can identify the most similar historical figures to any individual you query.

- We propose that information extracted from Wikipedia categories to be a reference standard to solve this task. Though not perfect as a detailed measurement, these human labeled features imply some relationships between similar people. We generated in total 3,000,000 triples of variable and prescribed difficulty, providing an effective standard to evaluate whether our distance measurement algorithms are reasonable.

- We compare two recent vector-based approaches with two traditional statistical methods. Unlike TF-IDF and LDA, vector based measurement (embedding) integrated distance function in its high dimensional representation. This feature make it easy to do operations like pairwise distance calculating, clustering and density estimation. All these approaches yield good qualities, but may focus on different aspects of similarity. We also generated a model using linear combination of previously mentioned models to get a better tradeoff between graph structures and text semantics.

- We collect data from 176 human volunteers for the ranking of 500 most famous people's Wikipedia categories. We adopt this data collection together with our similarity measurements to automatically rank the descriptive power of Wikipedia categories of historical figures. We

conduct a multi-level grid search to find the algorithm that can best quantify the importance of Wikipedia categories. We demonstrate that our ranking achieve an overall agreement of 75.41% with human voting on 5-choices questions.

6.1 Related work

Distributed word representations, or word embedding, are dense numerical representations of words that capture both semantic and syntactic features of words. Training word embedding needs only raw text corpus as input without human intervention or language dependent processing. The features embedding capture are task independent which make them ideal for language modeling. The original work for generating word embedding was presented in [10]. The embedding was a secondary output when generating language model.

There had been an interest in speeding up the generation process [11, 12] after [10]. Word embedding encodes many knowledge of languages and SENNA [26] showed that embedding are able to perform well on several NLP tasks in the absence of any other features. Huang et al. [45] applied local information can lead to better clustering of word embedding. Al-Rfou et al. [6] constructed word embedding for over 100 languages which provides even more resources to analyze text in many other languages.

DeepWalk is a novel approach for learning latent representations of vertices in a network [78]. It generalizes recent advancements in language modeling and unsupervised feature learning (or deep learning) from sequences of words to graphs. DeepWalk uses local information obtained from truncated random walks to learn latent representations by treating walks as the equivalent of sentences. These latent representations encode social relations in a continuous vector space, which is easily exploited by statistical models. Applying DeepWalk to Wikipedia will create embedding for each page in this huge graph thus provides statistical comparison between pages.

Latent Dirichlet allocation (LDA) is a generative model [14] and people can estimate the probability distribution of topics of a certain document. Krestel et al. [58] demonstrated good performances of LDA in tag recommendation – LDA showed its excellence via the fact that the topics highly agree with real tags when finding most important feature words of a page. However, LDA still has some defects. One is that the probability distribution

is not deterministic, especially when there are many related topics thus it is hard to measure similarities using LDA only, the other is that LDA focus on co-occurrence of words and semantic grouping of topics needs language dependent resources, like synonyms [34]. We will show later that models using word embedding performs as well as LDA when grouping topics.

All these method show some kinds of similarity measurement of pages based on text (or corresponding links), but how well they can estimate peoples' views on historical figures has not be analyzed.

6.2 Data collection

We start the whole corpus processing from English Wikipedia dump. First of all, we will check if a page is either a "Redirect" or a "Disambiguation" page. These two kinds of pages are designed to resolve alias of pages. They contain no text content but only redirection links to other pages. We ignore these two types of pages and then split each page into two different parts: content part and reference part. Content part contains main body of a Wikipedia page with markup and reference part includes Wikipedia category information, reference links and supplemental materials.

Before we remove Wikipedia markups to get raw text, we need links between Wikipedia pages to create a huge adjacent list for all Wikipedia article pages, recording whether there is a directed link from one page to another. We only use links in content part of each page without looking at references and categories links. This procedure helps us collect adjacent list of totally 4,517,721 pages, occupying 0.5GB of disk space.

Next we link each wiki page to corresponding freebase page and check if it falls into the category of a person. In order to make reasonable similarity, we ignore pages that have less than 50 hits which may probably point to insignificant people with very short introduction. We parse and tokenize all pages pointing to a person and keep raw text in content part without references or categories. We finally record 557,965 people's Wikipedia text, approximately occupying a size of 1.7GB in our database. Later on we will these Wikipedia texts to refer to corresponding people when calculate similarities.

After that we extract category information for these 557,965 people from reference part of their Wikipedia pages, which will be used to generate testing bed based these "category description" of a person. We eliminated trivial categories that makes no sense in comparing similarity, including "year of

birth", "year of death" and "living or not" using regular expressions and created a hash table to store these category information. This category information for people is about 0.15GB.

At last we intersect the raw text of people with SENNA's dictionary (which consists of 130,000 words) and tokenized page contents to build a word-frequency table for each word and document which will be used later in our models. We build these corpus, dictionary and TF-IDF using Gensim topic modeling tools [83]. We use a blacklist to get rid of common stop words and numbers in this step.

6.3 Model description

In this section we will describe four candidate vector models that can convert a Wikipedia page, in our case neutral description of one's history, to feature vectors to fuel similarity measurements. These models are word level TF-IDF, distributed word embedding, LDA topic modeling and Deepwalk embedding.

6.3.1 TF-IDF model

TF-IDF is frequently used in natural language processing to reflect how important a word is to a document in a collection or corpus. TF-IDF can easily remove common words with less descriptive power. Each page in our experiment will be converted into feature vector of size $\|V\|$ where V is the vocabulary that ever appear. We fill in 50 non-zero TF-IDF values in this vector indicating top 50 description words with highest TF-IDF, discarding the rest of "long tails". Similarity between two people could then be measured according to the distance, either L1 or L2 normalization, between corresponding feature vectors.

TF-IDF model includes all possible words in its vector space, which makes it useful to emphasize rare words that do not appear in SENNA's dictionary.

6.3.2 Distributed word embedding model

Distributed word embedding model is a simple "composite" idea that create articles' embedding using words' embedding. We apply a weighted average model to average SENNA's word representation of each word in the article

using their TF-IDF value as weights. Embedding of person will be close to the most important words in corresponding article and provides a good estimate of similarity between people since word embedding group similar words together. Figure 6.2 is a simplified projection illustrating separated apart distribution of party leaders, musicians and physicists in our embedding space.

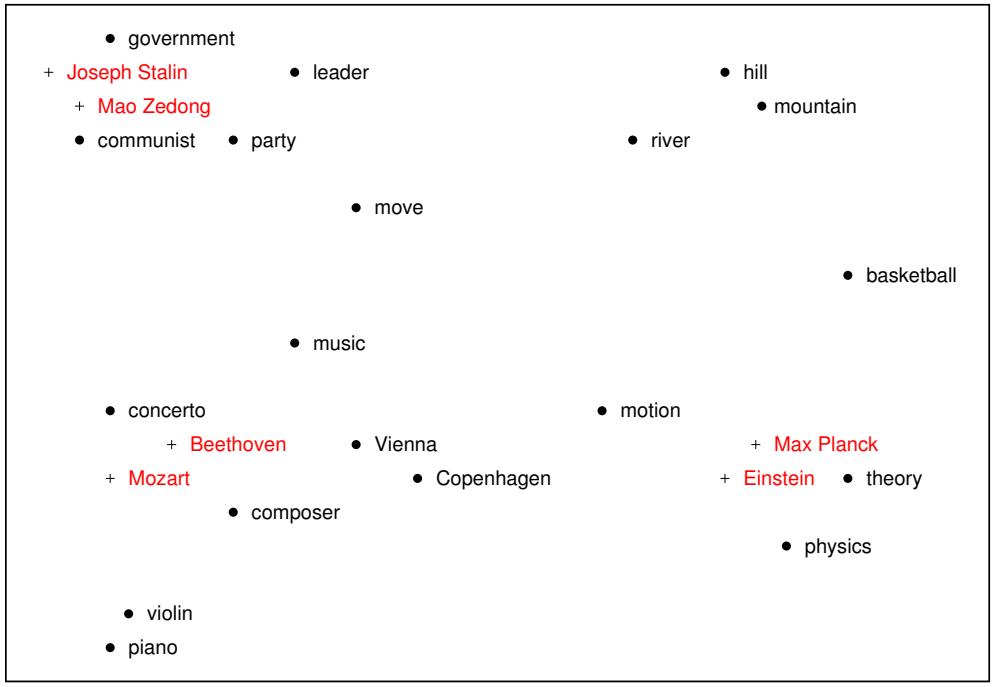


Figure 6.2: Sample entities and words example in projected embedding space, distance between pairs shows their relevance. Entities will be attracted by the most descriptive words and locate close to the "centroid".

Embedding of the article has same dimension as embedding of word in SENNA, which is 50. In our experiment distance function could be either Euclidean distance or Manhattan distance.

Distributed word embedding model embedding would be a reasonable extension of TF-IDF model since similar words in embedding spaces are grouped together. It makes full use of close neighbors in embedding space as synonyms thus greatly reduce the number of dimensions. However, dis-

tributed word embedding can only handle words with an existing embedding – any words that does not appear in SENNA's dictionary will be discarded.

6.3.3 LDA model

LDA is a mature model providing probability distributions of topics. It is based on co-occurrence of different words. Figure 6.3 shows an example of top related topics for some entities.

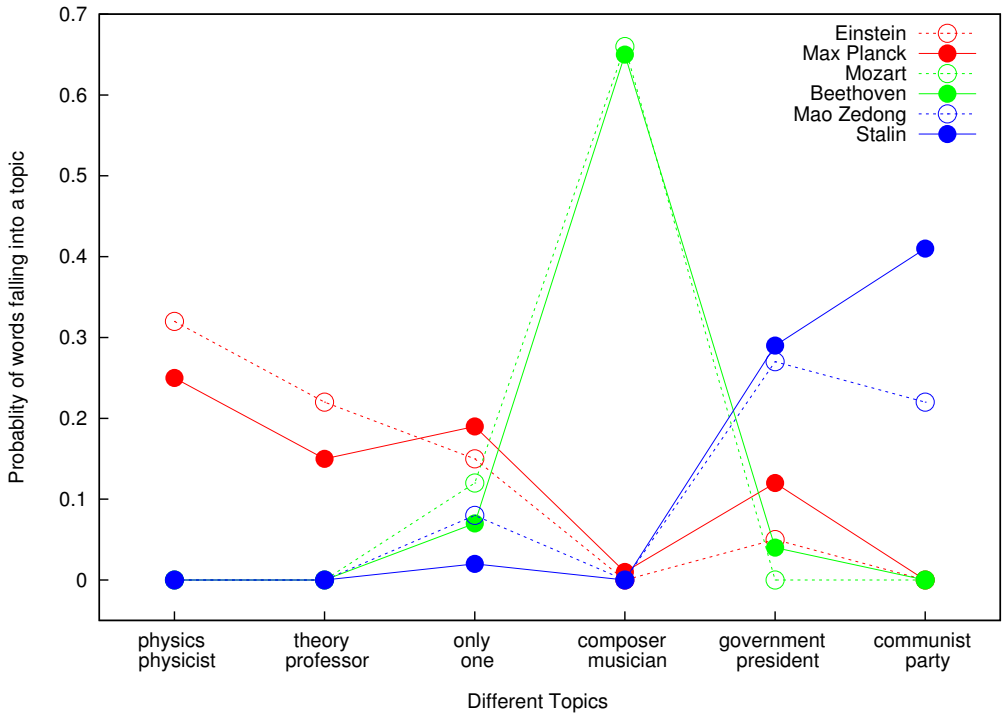


Figure 6.3: Examples of entities and top six related topics in LDA method. We demonstrate each topic by showing representative words. The probability of topic distribution differ a lot for party leaders, musician and physicists.

We convert each page in the corpus into a probability distribution of 500 possible topics. Pages have higher probability to fall into same topics should be considered more similar.

Distance used to compare topic distribution similarity could be Euclidean distance or Manhattan distance since we consider them to be a feature vector.

However, as a probability distribution we can also apply Jensen–Shannon divergence between LDA vectors in our experiment.

The advantage of LDA model is that output of each topic directly connect with other natural language processing tasks, thus it is easy for human beings to read and understand.

6.3.4 Deepwalk embedding model

Deepwalk is an online algorithm that creates statistical representation of graph by learning via random walks in the graph. Walks are considered as sentences metaphor and generate latent dimensions according to adjacent list. Since Deepwalk does not use entire graph at once, it can handle huge graphs of Wikipedia scale. With a hierarchical Softmax layer these latent dimensions will be finally converted into vector representations. In Wikipedia, pages sharing more links will sit closer in our Deepwalk embedding space. If a page has too many outgoing links, its connected page will have lower chance to be visited during random walk thus reduce weights of such links. We use the package described in [78] and create 128-dimension feature vector for each page in Wikipedia. Distance between two people will be set to either L1 or L2 normalization between embedding of corresponding pages. The final embedding size of all Wikipedia pages (in total 4,517,721) is 5.5GB and that for 557, 965 people is only 0.68GB.

Deepwalk fully utilizes connections in Wikipedia and train embedding of an article using it local connections in the huge adjacent graph. However, Deepwalk does not process of word itself. It might cause some disadvantages in semantically locating highly correlated words and further understanding the paragraph.

6.4 Wikipedia categories processing

We use extracted Wikipedia categories information to create a reference standard of "similarity" between people. Wikipedia categories are human annotations that intended to group together pages on similar subjects. Existing categories group people in many aspects, for example, nationality, job titles, awards, education histories. These features shows a reasonable part in our memory to remember people which appears to be a good representation of "similarities" in real life. From our observation, most categories

have strong signals of categorizing people, like "Nobel laureates in Physics", "Presidents of the United States"; some categories summarize interesting background knowledge but not as powerful as we expected, like "Austrian Roman Catholics", "People from Newark, New Jersey"; some categories are just too trivial to provide supporting evidences in our task, like "Living people".

We believe the importance of fame. Sometimes people with different fame level are not comparable – even if they exactly live a very similar life style. Historical figures with unbeatable contribution to their field are much easier to establish analogies with people in other fields, for instance, "Gary Larson is the Jimi Hendrix of comics". Since Wikipedia text does not provide the quantification of the fame feature, we turn into other sources to work as a tradeoff.

We show in Figure 6.4 the distribution of Wikipedia categories of people. "Famous level" is measured according to "ranking of fame" described in [88] which is a combination of Wikipedia hits, article length and links. The statistic shows that most people do not have many representative categories, which indicates that we cannot resolve our task totally based on Wikipedia categories. However, existing categories act as good summary of a person. The most famous 10,000 people have 20 categories on average and 14.08% of them occupying more than 30 categories. The most famous 50,000 people usually have 13 categories. Remaining people who are not that famous often capture 4 – 6 categories.

6.4.1 Constructing reference standard

We build up our reference standard based on the assumption that "the more common categories shared between two people, the more similar they are". However, not all these categories are useful. The "living people" category exists to help wiki editors improve the quality of biographies of living persons by ensuring that the articles maintain a properly sourced neutral point of view to protect them from inappropriate information. It is definitely meaningless to consider "living people" when performing an impressive analogy. We did a rough cleaning step to eliminate categories that are too broad to be representative in substantial similarity test like "living people" as well as categories indicating people who were born or died in a given year or location.

Let $F(X, Y)$ be the number of shared categories between person X and person Y in Wikipedia. We can calculate $F(X, Y)$ for any pair (X, Y) in

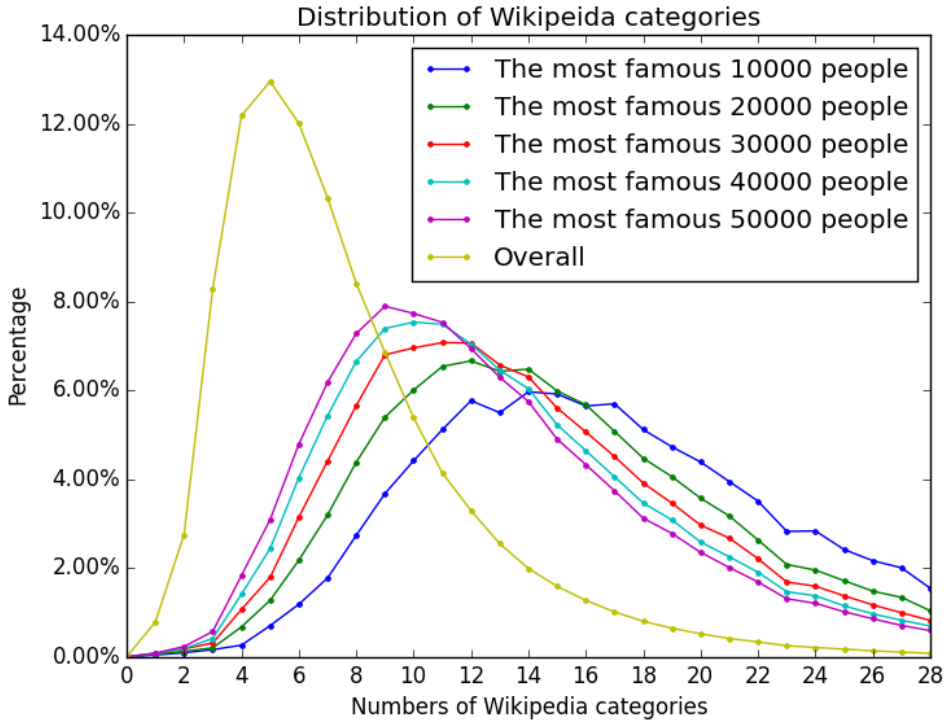


Figure 6.4: Distribution of Wikipedia category numbers on pages of people. It is clear that we usually have more detailed information on famous historical figures and the category comparison could be more precise. More famous people usually occupy more Wikipedia categories. Overall average categories between people lies between 8 and 9.

our database. In the next step we randomly select tuples of people (X, Y, Z) from our database such that $F(X, Y) > F(X, Z)$, which means X shares more common Wikipedia categories with Y than with Z, indicating Y is more similar to X than Z is. According to distribution of Wikipedia category numbers, $F(X, Y) > 3$ usually indicates a high similarity thus we consider pairs (X, Y) to have high (or low) similarity if $F(X, Y) > 3$ (or $0 < F(X, Y) \leq 3$). We sample 500,000 tuples of (X, Y, Z) in each of these 3 cases, for both all-people-tests and most-famous-people tests, sorted in difficulty level:

- **Case I: High similarity VS Zero**: $F(X, Y) > 3$ and $F(X, Z) = 0$

- **Case II: Low similarity VS Zero**: $0 < F(X, Y) \leq 3$ and $F(X, Z) = 0$
- **Case III: High similarity VS Low similarity**: $F(X, Y) > 3$ and $0 < F(X, Z) \leq 3$ and $F(X, Y) - F(X, Z) \geq 2$

Case	(X, Y, Z)
I	(Einstein, Aristotle, Celine Dion) (Lincoln, Bill Clinton, Heath Ledger) (Mozart, Charlie Chaplin, Larry Bird) (Mao Zedong, W. Churchill, Marco Polo) (Jesus, Isaac Newton, Sun Yat-sen)
II	(Einstein, Oppenheimer, Michael Jackson) (Lincoln, Reagan, Gaddafi) (Mozart, Brahms, Michael Phelps) (Mao Zedong, Deng Xiaoping, Lady Gaga) (Jesus, Moses, Kim Jong-il)
III	(Einstein, Richard Feynman, Hawking) (Lincoln, Ulysses Grant, George W. Bush) (Mozart, Beethoven, Dmitri Shostakovich) (Mao Zedong, Joseph Stalin, Bruce Lee) (Jesus, John the Baptist, Homer)

Table 6.1: Examples showing three cases of comparing different similarity levels. (X, Y) is always closer than (X, Z) based on counts of common Wikipedia categories. The accuracy will be judged by how well we recover such comparisons from our vectors.

To evaluate our distance measurements on these 3 cases shown in 6.1, we calculate for each tuple (X, Y, Z) the distance between X, Y as $Dist(X, Y)$ and the distance between X, Z as $Dist(X, Z)$. If $Dist(X, Y)$ and $Dist(X, Z)$ indicates that X is more similar to Y than Z, it agrees with what the tuple implies and we answer this tuple-based query correctly. The final performance of each measurement is reported as the percentage of correct tuples among 3 test cases as well as the overall percentages. As previously mentioned, $Dist$ function would be the Euclidean distance when measuring distance between points in embedding space (i.e. the Average embedding model) and JS divergence (Jensen–Shannon divergence) for probability distributions.

6.5 Which feature vector is better?

Table 6.2 listed accuracy of 4 candidate models with different parameters on test of all people in Wikipedia and on test of 50,000 most famous people.

Overall					
Model	Parameters	Case I	Case II	Case III	Overall
Random	N/A	50.00%	50.00%	50.00%	50.00%
TF-IDF	L2	85.89%	76.11%	77.87%	79.96%
	L1	85.54%	75.72%	77.33%	79.53%
Word embedding	L2	96.95%	84.40%	74.97%	85.44%
	L1	96.56%	84.26%	75.57%	85.46%
LDA	L2	98.70%	88.22%	75.39%	87.43 %
	L1	98.17%	88.35%	77.26%	87.92 %
	JS	97.69%	87.98 %	76.22%	87.29 %
Deepwalk	L2	**99.51%**	**89.50%**	**84.97%**	**91.33%**
	L1	99.11%	89.13%	84.59%	90.98%
Most Famous 50,000 people					
Model	Parameters	Case I	Case II	Case III	Overall
Random	N/A	50.00%	50.00%	50.00%	50.00%
TF-IDF	L2	87.75%	76.93%	78.24%	80.97%
	L1	86.59%	75.38%	77.01%	78.94%
Word embedding	L2	96.89%	82.92%	90.23%	90.01%
	L1	96.51%	82.90%	89.04%	89.48 %
LDA	L2	97.68%	83.71%	80.99%	87.46%
	L1	97.95%	83.31%	81.26%	87.51%
	JS	97.15%	83.89 %	81.14%	87.40%
Deepwalk	L2	**98.73%**	**85.47%**	**91.59%**	**91.93%**
	L1	98.11%	84.85%	90.79%	91.24%

Table 6.2: Accuracy performance of candidate models with different parameters.

TF-IDF model is undoubtedly better than Random guess and it can answer approximately 4 out of 5 questions correctly in general, which shows that a quick glancing (only top 50 words with highest TF-IDF) at the distribution of words in corpus can very well help finding the topic of an article.

However, TF-IDF do not consider any syntactic changes of words (e.g. great vs greatest) and synonyms (e.g. emperor vs monarch) so that it is hard to capture similar semantic meanings behind different words. Plus, study on error cases shows that TF-IDF focus too much on locations and names (e.g. the last name of James Simons ranked highly in TF-IDF), which reduces the ability to recognize more important words in similarity measurement.

Word embedding model solved some problems we have in TF-IDF models. By applying distributed word embedding, words with similar syntactic and semantic meanings will be clustered together and thus affect the embedding of the article in a similar way. We also see that enabling TF-IDF weight slightly increase the performance of DWE model from our experiments. This shows that the procedure of averaging embedding itself may already encoded frequency so that assigning rare words a higher weight cannot change much on distinguishing entities. However, getting embedding for phrases and articles still seem to be an interesting open question. Besides weighted average we have tried, there might also be other ways of converting word embedding to article embedding – an interesting idea was raised [61] which shows good performance on creating embedding of articles but whether it is not cheap enough to run on Wikipedia scale graph.

LDA model is very close to our Deepwalk embedding model in Case I and Case II. However, it performs badly in Case III, where three entities all share some similarities in topics. This might be fixed by a replacing Manhattan distance with a better measurement (e.g. JS-diversion) but it will be way expensive to perform on a Wikipedia scale graph. We also discovered that increasing number of topics in LDA model does not really boost the accuracy, which implies that topics itself is not strong enough comparing to semantics in the article in our task of identifying analogous historical figures. What's more, we discovered that too detailed topics might not benefit similarity measurement. For instance, Yao Ming could be tagged as "the famous Chinese basketball player in NBA" and Jeremy Lin could be, generally speaking, a good match of Yao Ming. However, for those who familiar with NBA, they play totally different positions and they have different playing style. This phenomenon indicates that we may not need that many dimensions or topics to decide features of a person in our task.

Deepwalk embedding yielded an overall accuracy of 91.3% on all people test and an even higher 91.93% on 50,000 most famous people test, winning all other vector based model. One noticeable drop in Case II accuracy shows that famous people will have more "weak" categories that do not provide

strong support on similarity measurement. Another discovery is that important historical event will usually pull people closer in Deepwalk embedding space, for instance, Ward Hill Lamon is considered close enough to Abraham Lincoln due to the famous assassination. Lamon was actually not any kind of politician which shows that Deepwalk embedding focuses more on features of co-working and it actually use little text information.

As we know, this accuracy table does not show everything we care about "similarity". For instance, "Genghis Khan" and "Napoleon" would be somewhat similar due to their military achievements as conquerors and their significance in history as monarchs. But these two people lived in different ages, rose from different locations and speak different languages thus it is hard to have a path of links connecting them through Wikipedia. Our program is able to find such relationships but the rank of "Napoleon" when querying "Genghis Khan" is still lower than expected.

No doubt that fames of a person should be considered part of similarity, so we did some combined experiments using Deepwalk embedding and significant score of people introduced in [88]. Combining Deepwalk model with significant score and LDA model which captures some interesting relationship in historical behavior through text, we gladly found examples that "Joseph Stalin" popping out to be the top 1 candidates when querying "Adolf Hitler". Table 6.3 shows some examples of finding analogous historical figures using single or combined distance measurement from our previous experiments.

6.6 Descriptive power of Wikipedia categories

In previous section, we use count of shared Wikipedia categories as a reference standard of measuring similarity between historical people. However, it is clear that not all categories are created with equal descriptive power. For instance, "Presidents of the U.S. " are leaders of the most powerful country in the world and people in this category definitely have more impact than local senators or state lawyers; categories being abandoned like "year of births" and "year of deaths" in previous experiment usually makes no sense. We seek to quantify the importance of Wikipedia categories using feature vectors of 557,596 people we have generated and propose an algorithm to quantify and rank these categories.

Model	Albert Einstein			Yao Ming			Larry Page		
	Candidates	C	HE	Candidates	C	HE	Candidates	C	HE
Word embedding	Max Planck	3	GOOD	Yi Jianlian	4	GOOD	Alan Kotok	0	OK
	Ernst Mach	2	OK	Luol Deng	1	OK	Fred Brooks	1	OK
	Erwin Schrödinger	2	GOOD	Anthony Parker	0	OK	Bob Wallace	0	BAD
	Arthur Eddington	0	OK	Andrés Nocioni	2	OK	Robert Metcalfe	1	GOOD
	Richard Feynman	4	GOOD	James Yap	0	BAD	R. P. Gabriel	2	GOOD
	Paul Dirac	2	GOOD	Mengke Bateer	5	GOOD	Eric Eldred	0	BAD
	Freeman Dyson	3	OK	Vijay Singh	0	BAD	Brendan Kehoe	0	BAD
	Norbert Wiener	1	BAD	C-M Wang	0	BAD	Ted Nelson	0	OK
	Georges Lemaître	1	BAD	Yu Darvish	0	BAD	Donald Davies	0	BAD
	Enrico Fermi	3	GOOD	Chris Bosh	2	GOOD	M. Stachowiak	0	OK
LDA	Wolfgang Pauli	5	GOOD	Yi Jianlian	4	GOOD	Simson Garfinkel	0	GOOD
	Emmy Noether	1	BAD	Chris Bosh	2	GOOD	Robert Metcalfe	1	GOOD
	Erwin Schrödinger	2	GOOD	Sun Yue	3	GOOD	John Mashey	0	BAD
	Eugene Wigner	4	GOOD	Luol Deng	1	OK	Ray Tomlinson	0	BAD
	Norbert Wiener	1	BAD	Bob Cousy	1	BAD	M. J. Dominus	0	BAD
	Esther Lederberg	0	BAD	Steve Nash	2	OK	R. Piquepaille	0	BAD
	David Hilbert	0	OK	Herschel Walker	0	BAD	Ellen Spertus	1	BAD
	Felix Ehrenhaft	0	OK	R. Tomjanovich	2	OK	Jon Lebkowsky	0	OK
	Paul Ehrenfest	1	OK	A. Kavaliauskas	1	OK	R. P. Garbriel	2	GOOD
	Ralph Kronig	1	OK	Mengke Bateer	5	GOOD	D. Giampaolo	1	OK
Deepwalk	Richard Feynman	4	GOOD	Yi Jianlian	4	GOOD	Sergey Brin	12	GOOD
	Max Planck	3	GOOD	Jeremy Lin	0	GOOD	Eric Schmidt	6	GOOD
	Freeman Dyson	3	OK	Kobe Bryant	2	OK	Bill Gates	6	GOOD
	David Bohm	1	OK	Wang Zhizhi	5	GOOD	Marc Andreessen	2	GOOD
	Stephen Hawking	2	GOOD	Michael Jordan	1	OK	Mark Zuckerberg	6	GOOD
	David Hilbert	0	OK	Deron Williams	2	OK	Esther Dyson	0	BAD
	Oppenheimer	4	GOOD	Mengke Bateer	5	GOOD	John Doerr	3	OK
	Werner Heisenberg	4	GOOD	Dwyane Wade	3	OK	John Battelle	0	GOOD
	Hermann Bondi	1	BAD	LeBron James	3	OK	Joi Ito	0	BAD
	Erwin Schrödinger	2	GOOD	Steve Francis	2	OK	Jimmy Wales	1	OK
Linear comb of Deepwalk and LDA	Max Planck	3	GOOD	Yi Jianlian	4	GOOD	Bill Gates	6	GOOD
	Erwin Schrödinger	2	GOOD	Mengke Bateer	5	GOOD	Eric Schmidt	6	GOOD
	Richard Feynman	4	GOOD	Chris Bosh	2	GOOD	Simson Garfinkel	0	GOOD
	Freeman Dyson	3	OK	Michael Jordan	1	OK	Sergey Brin	12	GOOD
	Wolfgang Pauli	5	GOOD	LeBron James	3	OK	Robert Metcalfe	1	GOOD
	David Bohm	1	OK	Jeremy Lin	0	GOOD	Marc Andreessen	2	GOOD
	Eugene Wigner	4	GOOD	Charles Barkley	2	GOOD	Mark Zuckerberg	6	GOOD
	R. Millikan	2	OK	Tony Parker	1	OK	John Battelle	0	GOOD
	Stephen Hawking	2	GOOD	Steve Francis	2	OK	Marissa Mayer	4	OK
	George Gamow	2	OK	Juwan Howard	2	GOOD	Steve Jobs	5	GOOD

Table 6.3: Examples of 10 closest neighbors we find using our vector based models and comparison to our reference standard. *C* column represents count of common Wikipedia categories between pairs of people and *HE* column shows human evaluation after reading their bibliography, if it is a GOOD, OK or BAD match according to general knowledge.

6.6.1 Collecting human annotations

We start a project on Crowdflower, a leading people-powered data enrichment platform, to collect gold standard of the importance level of Wikipedia

categories. We pick 500 most famous people according to the fame ranking in our previous experiment to collect information from human volunteers. For each Wikipedia person we manually select 4 categories which are good enough to make description, plus 1 random category to make up a question with 5 choices. Each volunteer is required to pick the most important and descriptive one among these 5 choices as shown in Figure 6.5.

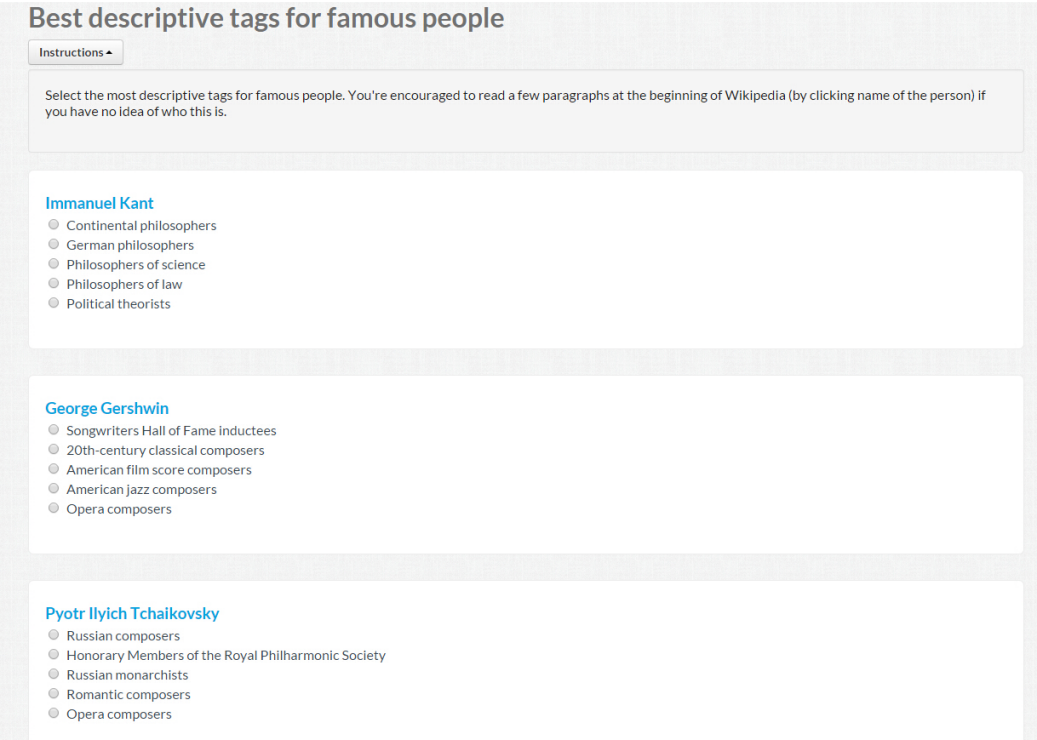


Figure 6.5: Question samples to collect human wisdom. For each question we collect 20 answers and the distribution of these answers indicate overall importance of each category.

We collect 10 answers for each question regarding a Wikipedia person. In total we gathered 5,000 answers from 176 volunteers, covering 1076 most important categories. Each answer makes a clarification that one choice is dominating the other four. We assume more descriptive categories will have more votes and the distribution of these votes implies importance of each Wikipedia categories.

Our observations on the data collection show that some correlative cat-

egories are highly confusing as they often co-appear and they share similar descriptive power. Table 6.4 lists top 10 most confusing category pairs.

Category pairs	Co-Prob
French Open champions Wimbledon champions	100.00%
Australian film actors Australian television actors	100.00%
Association football forwards Brazilian footballers	86.60%
Holocaust perpetrators Nazi Germany ministers	86.60%
National Basketball Association All-Stars Parade High School All-Americans (boys' basketball)	81.65%
Eastern Orthodox saints People celebrated in the Lutheran liturgical calendar	75.59%
American novelists American short story writers	67.94%
American jazz singers Traditional pop music singers	67.61%
American rhythm and blues singer-songwriters American soul singers	67.36%
African-American rappers Pseudonymous rappers	62.90%

Table 6.4: 10 most confusing category pairs in our questions. *Co-Prob* of two categories A and B is defined as the geometric mean of $P(A|B)$ and $P(B|A)$. Since candidate choices are manually picked from existing Wikipedia categories, such co-occurrence reflect some level of general human background knowledge preference.

Since the table is made based only on the categories of 500 most famous people, it is not accurate enough to measure the co-occurrences of all category pairs. However, it points out that people have high tolerances on categories with similar importance level.

6.6.2 Definition of close neighbors

We demonstrated in previous section that distance between our feature vectors indicate similarity between corresponding Wikipedia people. However,

distance in embedding space is not linear – pairs with twice the distance does not necessarily mean half the similarity. On the other hand, observation on word embedding shows that only pairs within a certain range show signals of similarity. This might still be true for our feature vectors on Wikipedia people. Such ranges are not pre-defined and usually it is correlated with the density of embedding in a certain area of the embedding space.

We propose two basic approaches. One is to limit close neighbors by count, considering the closest K neighbors in embedding space to be close neighbors. Since relationship of close neighbors is not always reversible under this definition, this strategy will usually create asymmetric results. The other is to pick close neighbors by distance, marking all neighbors within a certain distance of D as close. With this strategy, if a point in space is semi-isolated then nothing would be considered similar to it. Both two approaches are reasonable and will be considered as a hyper-parameter in our experiment.

6.6.3 Ranking a category

We propose two methods to quantify how important a Wikipedia category is when describing people.

The first one measures "tightness" of a category, which is defined to be the ratio of close neighbors inside category and close neighbors outside category. This is a simple and direct measurement because category with more inside close neighbors will be more stable and more likely to become close neighbors and sharing similarities with "members inside this category". However, this method does not consider the size of the category well. For larger categories with more corresponding people, it is even harder to maintain all close neighbors to be members inside the category. More points on the high-dimensional convex hull will establish close neighbor relationships with points outside the category, thus reducing the value of "tightness".

Let T_{cat} be the tightness of category, then we have:

$$T_{cat} = \sum_{\substack{X \in cat \\ Y \in cat \\ (X,Y) \in closeneighbors}} / \sum_{\substack{X \in cat \\ (X,Y) \in closeneighbors}}$$

The other measurement focuses on balancing the effect of both size of the category and the probability of inside-category pairs become close neighbors. We assume that the probability of "a pair is inside a category" is independent

from the probability of "a pair qualify as close neighbors". Consider a point A and the probability of its close neighbor B is also a member of the category will be:

$$P_{cat} = \frac{\sum_{X \in cat}}{\sum_{X}}$$

If we randomly pick C close neighbors, the probability of having K close neighbors in the group will be:

$$P(cat, K, C) = \binom{C}{K} ({P_{cat}}^{K} * (1 - P_{cat})^{C-K})$$

We then count the average number of close neighbors G from observations in embedding space and the surprise level of the category will be defined as:

$$S_{cat,C} = \sum_{X \geq G} P(cat, G, C)$$

Here C will set to a fixed number 100 as a hyper parameter.

Both measurement of tightness and surprise level will be experimented to find the best agreement with human annotations.

6.7 Evaluating category ranking

We then conduct a grid search with following choices:

- Feature vectors: (TF-IDF, Word embedding, LDA, Deepwalk)
- Distance function: (L1, L2 and JS for LDA)
- Close neighbors: (Count and Distance). We will test on cases of average close neighbors of 5, 10, 25, 50, 100 and corresponding distance that keep the same number of close neighbors.
- Measurement: (Tightness and Surprise level)

Final quality will be evaluated according to the agreement with human annotations, i.e. probability of agreeing with top voted choice as well as the second, third, fourth, last choices.

Since each vote provides 4 comparisons, agreeing with voting for i-th choice means we are making (5 – i) correct judgements, thus we set up an overall accuracy:

$$Overall = \sum_{i=1}^{5} Agreement\ with\ i^{th}\ vote * (5–i) * 0.25$$

Since there exist confusing category pairs, the best category is not always dominating. The best ranking constructed using topological order of the gold standard vote can achieve an overall accuracy of 84.56%.

All experiments are conducted on 50,000 most famous people to avoid meaningless comparisons.

6.7.1 Influence of feature vectors

We first discuss the influence of four different feature vectors in Table 6.5.

Feature vector	Best performance	Agreements				
		1st	2nd	3rd	4th	5th
TF-IDF	72.49%	43.71%	24.73%	14.25%	12.45%	4.87%
Word embedding	71.68%	42.13%	24.23%	17.40%	10.71%	5.54%
LDA	73.31%	44.07%	25.10%	16.29%	9.11%	5.44%
Deepwalk	**74.64%**	46.47%	24.79%	14.97%	8.32%	5.44%

Feature vector	Parameters	Corresponding Distance				
		5	10	25	50	100
TF-IDF	(Count=25, L1, Surprise)	3.3716	3.5729	3.9743	4.1376	4.4433
Word embedding	(Count=25, L2, Surprise)	0.4770	0.5046	0.5463	0.5828	0.6255
LDA	(Count=25, L1, Surprise)	0.1352	0.1972	0.3091	0.4223	0.5639
Deepwalk	(Count=50, L2, Surprise)	1.0092	1.0704	1.1740	1.2823	1.4366

Table 6.5: Best performance of each feature vectors. Deepwalk outperforms the others and achieve 74.64% overall accuracy and 46.47% agreement with human top votes. *Corresponding distance* gives the threshold of distance to guarantee a certain average of close neighbors for overall.

The rank of embedding basically follow the same order of previous similarity test. Distance distribution of TF-IDF and Word embedding is less reasonable – the gap between each threshold is linear but the count of close neighbors included is doubled, which suggests that close neighbors definitions in LDA and Deepwalk are more stable. Overall Deepwalk works slightly better than LDA. Considering the best performance to be 84.56% under out experiment, Deepwalk actually correcting 15.73% wrong answers. Agreement with 1st vote from human is also much higher.

6.7.2 Influence of distance measurement

Table 6.6 shows statistics of how distance measurement changes the performance.

Feature vector	Distance	Best Overall	1st	2nd	3rd	4th	5th
TF-IDF	L2	71.85%	42.17%	25.12%	15.84%	11.70%	5.17%
	L1	**72.49%**	43.71%	24.73%	14.25%	12.45%	4.87%
Word embedding	L2	**71.68%**	42.13%	24.23%	17.40%	10.71%	5.54%
	L1	71.32%	41.16%	24.35%	18.31%	10.97%	5.21%
LDA	L2	73.25%	44.39%	24.03%	16.67%	10.02%	4.89%
	L1	**73.31%**	44.07%	25.10%	16.29%	9.11%	5.44%
	JS	73.10%	43.99%	24.12%	17.33%	9.41%	5.14%
Deepwalk	L2	**74.64%**	46.47%	24.79%	14.97%	8.32%	5.44%
	L1	74.37%	46.09%	24.91%	14.91%	8.57%	5.52%

Table 6.6: Best performance of each feature vectors and distance measurement combinations. *Best Overall* shows the best performance achievable and the last 5 columns demonstrate per-rank agreement. There are no big gaps between L1 normalization and L2 normalization for all four types of feature vectors. Also, JS divergence does not yield better performances.

There are no big gaps between L1 normalization and L2 normalization for all four types of feature vectors. Also, JS divergence does not yield better performances. This phenomena probably indicates the redundancy in word embedding so that no matter which normalization function was chosen, there will be a factor that preserve the property of making similar embedding close enough.

6.7.3 Influence of defining close neighbors

Figure 6.6 shows 10 different definitions of close neighbors with corresponding performances.

It is clear that judging close neighbors using only the value of distance is worse. The reason is that density of local surrounding of semi-isolated points (e.g. person with few introduction text and Wikipedia links) is much lower than those frequently mentioned historical figures. Setting threshold to be a certain diameter will create uneven distribution of close neighbors, thus lower quality and stability of similarity measurement.

On the other hand, both count and distance threshold shows a peak within the range of 25 to 50 average close neighbors, which is approximately 0.05% to 0.1% of the whole data collection. Neither increasing nor decreasing this

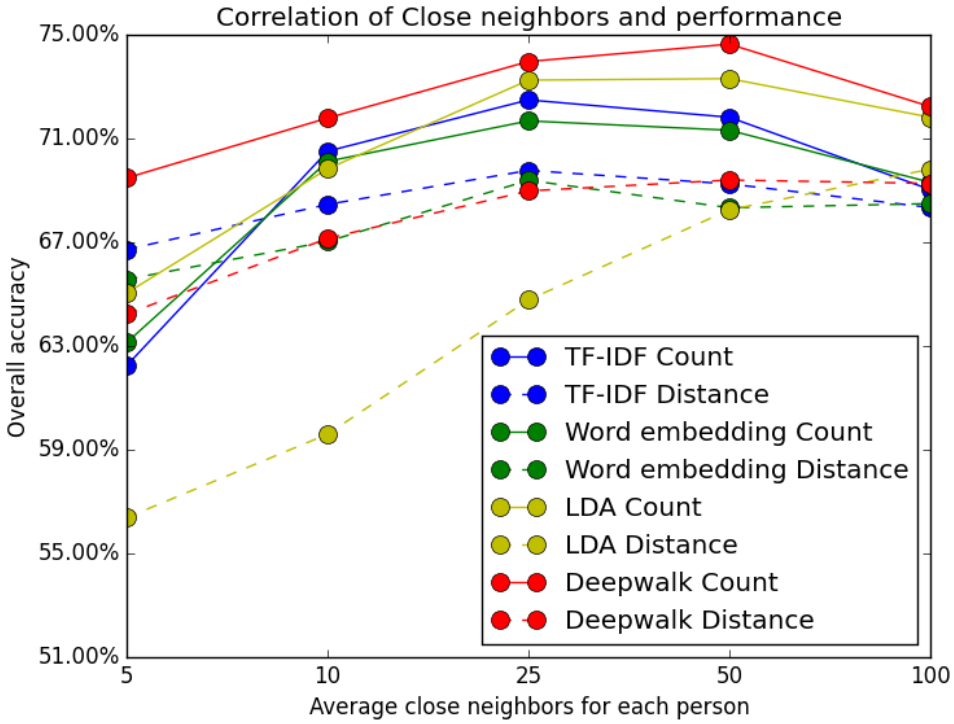


Figure 6.6: Different definition of close neighbors and corresponding best performance. We experimented strategies of making a fixed number of close neighbors for each point as well as creating a comparable number of close neighbors overall using a distance limitation. Fixed number strategy outperforms the distance definition.

value will give better performances. We believe such criteria would still work even under more sophisticated circumstance, e.g. considering all 557,596 people since points in the embedding space are usually evenly distributed.

6.7.4 Influence of importance measurement

Table 6.7 lists comparisons between "tightness" and "surprise level". One interesting observation is, the best performance of all Tightness measurement happens when creating an average of 100 close neighbors for each point in embedding space. However, we discovered that Surprise level measurement

still outperforms Tightness even with much fewer close neighbors, indicating that the size of category should be carefully examined during the procedure.

Feature vector	Measurement	Best performance	1st	2nd	3rd	4th	5th
TF-IDF	Tightness	68.22%	37.81%	23.18%	19.28%	13.56%	6.18%
	Surprise	**72.49%**	43.71%	24.73%	14.25%	12.45%	4.87%
Word embedding	Tightness	67.94%	37.77 %	23.20%	18.65%	13.82%	6.57%
	Surprise	**71.68%**	42.13%	24.23%	17.40%	10.71%	5.54%
LDA	Tightness	68.56%	36.49%	26.23%	19.24%	11.11%	6.93%
	Surprise	**73.31%**	44.07%	25.10%	16.29%	9.11%	5.44%
Deepwalk	Tightness	65.65%	34.51%	24.85%	18.49%	13.03%	9.11%
	Surprise	**74.64%**	46.47%	24.79%	14.97%	8.32%	5.44%

Table 6.7: Best performance of each feature vectors and importance level measurement. Surprise level measurement outperforms Tightness, indicating that the size of category should be carefully examined during the procedure.

6.8 Categories with bad performances

We try to make a deep analysis based on the best ranking, which is generated using Deepwalk embedding, L2 normalization, each person having 25 close neighbors and measured via Surprise level. We list all categories with more votes but less surprise level in our data collection. Table 6.8 shows the top 10 of them:

As we can see, 3 out of 10 categories (Prime Ministers of the United Kingdom, Presidents of the United States, First Ladies of the United States) are political leaders whose titles are so recognizable that they bestowed enough to distinguish this person from the others. However, such categories have a long history – it is quite possible that there are less similarity between U.S. presidents in 1800s and U.S. presidents after 2000 except the title itself and Deepwalk did not find supporting evidences from Wikipedia links. "The Beatles Members" plays a special role since the size of the category is too small while the descriptive power is unbelievably large. The remaining categories are rough and generalized, sometimes with confusions, which makes it hard to process.

Category	Count	Probability
Prime Ministers of the United Kingdom	162	85.26%
Presidents of the United States	116	82.86%
American pop singer-songwriters	111	58.42%
The Beatles members	80	72.73%
American rhythm and blues singers	75	62.50%
American male professional wrestlers	54	60.00%
First Ladies of the United States	52	86.67%
American rock singers	50	83.33%
American rock guitarists	50	71.43 %
American horror writers	12	62.50%

Table 6.8: Categories disagree most with votes. *Count* shows number of times human vote for this category but not higher-ranked ones. *Probability* shows the chance of this category being answered whenever it appears.

6.9 Conclusion

We have proposed models for constructing feature vectors and measuring the similarity between historical figures, and demonstrated that it works effectively over a representative evaluation environment. We tested our models on approximately 600,000 historical figures from Wikipedia pages, and investigate several approaches to similarity detection to uncover historical analogies. Our Deepwalk embedding of Wikipedia links yielded an overall accuracy of 91.3% in our evaluation and shows a good match of human annotated Wikipedia categories and a combination of our model can make query results even more reasonable.

These models naturally extend to analyzing figures in different languages, and also to extend to other classes of entities like locations (i.e. cities and countries) and organizations (companies and universities) and we are able to identify similar individuals for suggesting friends in social networks, or even matching algorithms pairing up roommates or those seeking romantic partners.

The final target of our application is not only focusing on identifying analogous historical figures on English version of Wikipedia. Considering possible applications of finding similar people via their personal webpage or resume (which focus on text) or their social network friends list (using graph structures), we are glad to see that our models can be applied to various types

of text and graphs and our models do not cost much for even Wikipedia scale corpus. Such utility could help improve algorithms of online recommending systems. With embedding of other language [6] we can even create an multilingual embedding space for people of different language backgrounds.

We are currently working to parameterize our methods so we can capture different tradeoffs between personality, temporal, and topic-based analogies. An inspection of our closest matches suggests that topic-based analogies dominate the nearest matches when considering text only, but more revealing analogies may result from restricting the analyzed word features to particular parts of speech or sentiment polarity.

Finally, we explain similarity more precisely using human-interpretable names of Wikipedia categories as dimensions/topics obtained using our learning procedures. We collect better knowledge to understand the importance of these particular strong or overrepresented features in our analysis. Our ranking of similarities provides excellent knowledge that can properly define weights of certain aspects to reduce the ambiguity of "similarity" and greatly improve the performance.

References

[1] Ahmed Abbasi, Hsinchun Chen, and Arab Salem. Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums. *ACM Transactions on Information Systems (TOIS)*, 26 (3):12, 2008.

[2] Muhammad Abdul-Mageed, Mona T Diab, and Mohammed Korayem. Subjectivity and sentiment analysis of modern standard arabic. In *ACL (Short Papers)*, pages 587–591, 2011.

[3] Nasreen AbdulJaleel and Leah S Larkey. Statistical transliteration for english-arabic cross language information retrieval. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 139–146. ACM, 2003.

[4] Khurshid Ahmad, David Cheng, and Yousif Almas. Multi-lingual sentiment analysis of financial news streams. In *Proc. of the 1st Intl. Conf. on Grid in Finance*, 2006.

[5] Yaser Al-Onaizan and Kevin Knight. Machine transliteration of names in arabic text. In *Proceedings of the ACL-02 workshop on Computational approaches to semitic languages*, pages 1–13. Association for Computational Linguistics, 2002.

[6] Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. Polyglot: Distributed word representations for multilingual nlp. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 183–192, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/W13-3520`.

[7] Carmen Banea, Rada Mihalcea, Janyce Wiebe, and Samer Hassan. Multilingual subjectivity analysis using machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 127–135. Association for Computational Linguistics, 2008.

[8] Valerio Basile and Malvina Nissim. Sentiment analysis on italian tweets. *WASSA 2013*, page 100, 2013.

[9] Mikhail Bautin, Lohit Vijayarenu, and Steven Skiena. International sentiment analysis for news and blogs. In *ICWSM*, 2008.

[10] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3: 1137–1155, 2003.

[11] Y. Bengio, J.S. Senécal, et al. Quick training of probabilistic neural nets by importance sampling. In *AISTATS Conference*, 2003.

[12] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.

[13] Shane Bergsma and Dekang Lin. Bootstrapping path-based pronoun resolution. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 33–40, Sydney, Australia, July 2006. Association for Computational Linguistics.

[14] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[15] Roger Boada, Rosa Sanchez-Casas, Jose M Gavilan, Jose E Garcia-Albea, and Natasha Tokowicz. Effect of multiple translations and cognate status on translation recognition performance of balanced bilinguals. *Bilingualism: Language and Cognition*, 16(01):183–197, 2013.

[16] Erik Boiy and Marie-Francine Moens. A machine learning approach to sentiment analysis in multilingual web texts. *Information retrieval*, 12 (5):526–558, 2009.

[17] Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58. Association for Computational Linguistics Baltimore, MD, USA, 2014.

[18] Chris Brew, David McKelvie, et al. Word-pair extraction for lexicography. In *Proceedings of the second international conference on new methods in language processing*, pages 45–55. Citeseer, 1996.

[19] Govenment Census. Genealogy data: Frequently occurring surnames from census 2000, `http://www.census.gov/genealogy/www/data/2000surnames/index.html`, 2000.

[20] Mauro Cettolo, Christian Girardi, and Marcello Federico. Wit3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, pages 261–268, 2012.

[21] Pedro J Chamizo Dominguez and Brigitte Nerlich. False friends: their origin and semantics in some selected languages. *Journal of Pragmatics*, 34(12):1833–1849, 2002.

[22] Yanqing Chen and Steven Skiena. Building sentiment lexicons for all major languages. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers)*, pages 383–389, 2014.

[23] Yanqing Chen, Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. The expressive power of word embeddings. *arXiv preprint arXiv:1301.3226*, 2013.

[24] P.C. Cohen. The emergence of numeracy. *Mathematics and democracy: The case for quantitative literacy*, pages 23–30, 2001.

[25] R. Collobert. Deep learning for efficient discriminative parsing. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[26] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

[27] D. Crayen and J. Baten. Global trends in numeracy 1820–1949 and its implications for long-term growth. *Explorations in Economic History*, 47(1):82–99, 2010.

[28] Paula Cristoffanini, Kim Kirsner, and Dan Milech. Bilingual lexical representation: The status of spanish-english cognates. *The Quarterly Journal of Experimental Psychology*, 38(3):367–393, 1986.

[29] Kerstin Denecke. Using sentiwordnet for multilingual sentiment analysis. In *Data Engineering Workshop, 2008. ICDEW 2008. IEEE 24th International Conference on*, pages 507–512. IEEE, 2008.

[30] Ton Dijkstra, Jonathan Grainger, and Walter JB Van Heuven. Recognition of cognates and interlingual homographs: The neglected role of phonology. *Journal of Memory and Language*, 41(4):496–518, 1999.

[31] Nadir Durrani, Hieu Hoang, Philipp Koehn, and Hassan Sajjad. Integrating an unsupervised transliteration model into statistical machine translation. *EACL 2014*, page 148, 2014.

[32] Esdict.com. False spanish english cognates, `http://www.esdict.com/downloads/false-spanish-english-cognates.pdf`, 2014.

[33] Andrea Esuli and Fabrizio Sebastiani. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, volume 6, pages 417–422, 2006.

[34] C. Fellbaum. Wordnet. *Theory and Applications of Ontology: Computer Applications*, pages 231–243, 2010.

[35] Christiane Fellbaum. *WordNet*. Wiley Online Library, 1999.

[36] June NP Francis, Janet PY Lam, and Jan Walls. The impact of linguistic differences on international brand name standardization: A comparison of english and chinese brand names of fortune-500 companies. *Journal of International Marketing*, 10(1):98–116, 2002.

[37] Wei Gao, Kam-Fai Wong, and Wai Lam. Phoneme-based transliteration of foreign names for oov problem. In *Natural Language Processing–IJCNLP 2004*, pages 110–119. Springer, 2005.

[38] Stefan Gindl, Albert Weichselbraun, and Arno Scharl. Cross-domain contextualisation of sentiment lexicons. *19th European Conference on Artificial Intelligence (ECAI)*, 2010.

[39] Alexandru-Lucian Gînscă, Emanuela Boroş, Adrian Iftene, Diana TrandabĂţ, Mihai Toader, Marius Corîci, Cenel-Augusto Perez, and Dan Cristea. Sentimatrix: multilingual sentiment analysis service. In *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*, pages 189–195. Association for Computational Linguistics, 2011.

[40] N. Godbole, M. Srinivasaiah, and S. Skiena. Large-scale sentiment analysis for news and blogs. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM)*, volume 2, 2007.

[41] Yulan He, Harith Alani, and Deyu Zhou. Exploring english lexicon knowledge for chinese sentiment analysis. *CIPS-SIGHAN Joint Conference on Chinese Language Processing*, 2010.

[42] Kanayama Hiroshi, Nasukawa Tetsuya, and Watanabe Hideo. Deeper sentiment analysis using machine translation technology. In *Proceedings of the 20th international conference on Computational Linguistics*, page 494. Association for Computational Linguistics, 2004.

[43] Gumwon Hong, Min-Jeong Kim, Do-Gil Lee, and Hae-Chang Rim. A hybrid approach to english-korean name transliteration. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*, pages 108–111. Association for Computational Linguistics, 2009.

[44] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM, 2004.

[45] Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of*

the Association for Computational Linguistics: Long Papers - Volume 1, ACL '12, pages 873–882, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=2390524.2390645`.

[46] F. Huang and A. Yates. Distributional representations for handling sparsity in supervised sequence-labeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 495–503. Association for Computational Linguistics, 2009.

[47] Diana Inkpen, Oana Frunza, and Grzegorz Kondrak. Automatic identification of cognates and false friends in french and english. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 251–257, 2005.

[48] Jagadeesh Jagarlamudi and Hal Daumé III. Regularized interlingual projections: evaluation on multilingual transliteration. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 12–23. Association for Computational Linguistics, 2012.

[49] Valentin Jijkoun, Maarten de Rijke, and Wouter Weerkamp. Generating focused topic-specific sentiment lexicons. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 585–594. Association for Computational Linguistics, 2010.

[50] Nobuhiro Kaji and Masaru Kitsuregawa. Building lexicon for sentiment analysis from massive collection of html documents. In *EMNLP-CoNLL*, pages 1075–1083, 2007.

[51] Amit Kirschenbaum and Shuly Wintner. Lightly supervised transliteration for machine translation. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 433–441. Association for Computational Linguistics, March 2009. URL `http://www.aclweb.org/anthology/E09-1050`.

[52] Amit Kirschenbaum and Shuly Wintner. A general method for creating a bilingual transliteration dictionary. In *Proceedings of the Seventh conference on International Language Resources and Evaluation*

(LREC'10), pages 273–276. European Language Resources Association (ELRA), May 2010. ISBN 2-9517408-6-7.

[53] Kevin Knight and Jonathan Graehl. Machine transliteration. *Computational Linguistics*, 24(4):599–612, 1998.

[54] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics, 2007.

[55] Peter Kolb. Disco: A multilingual database of distributionally similar words. *Proceedings of KONVENS-2008, Berlin*, 2008.

[56] Grzegorz Kondrak. Combining evidence in cognate identification. In *Advances in Artificial Intelligence*, pages 44–59. Springer, 2004.

[57] Grzegorz Kondrak and Bonnie Dorr. Identification of confusable drug names: A new approach and evaluation methodology. In *Proceedings of the 20th international conference on Computational Linguistics*, page 952. Association for Computational Linguistics, 2004.

[58] Ralf Krestel, Peter Fankhauser, and Wolfgang Nejdl. Latent dirichlet allocation for tag recommendation. In *Proceedings of the third ACM conference on Recommender systems*, pages 61–68. ACM, 2009.

[59] Vivek Kulkarni, Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. Statistically significant detection of linguistic change. *Proceedings of the 2015 WWW International Conference on World Wide Web, Florence, Italy*, 2015.

[60] A Kumaran, Mitesh M Khapra, and Haizhou Li. Whitepaper of news 2010 shared task on transliteration mining. In *Proceedings of the 2010 Named Entities Workshop*, pages 29–38. Association for Computational Linguistics, 2010.

[61] Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. *arXiv preprint arXiv:1405.4053*, 2014.

[62] Fangtao Li, Minlie Huang, and Xiaoyan Zhu. Sentiment analysis with global topics and local dependency. In *AAAI*, 2010.

[63] Words Worldwide Limited. Word list of us/uk spelling variants, `http://www.wordsworldwide.co.uk/docs/Words-Worldwide-Word-list-UK-US-2009.doc`, May 2009.

[64] Bing Liu. Sentiment analysis and subjectivity. *Handbook of natural language processing*, 2:568, 2010.

[65] Bing Liu. *Sentiment Analysis and Opinion Mining*. Morgan and Claypool, 2013.

[66] Gideon S Mann and David Yarowsky. Multipath translation lexicon induction via bridge languages. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, pages 1–8. Association for Computational Linguistics, 2001.

[67] David Matthews. Machine transliteration of proper names. *Master's Thesis, University of Edinburgh, Edinburgh, United Kingdom*, 2007.

[68] J.B. Michel, Y.K. Shen, A.P. Aiden, A. Veres, M.K. Gray, J.P. Pickett, D. Hoiberg, D. Clancy, P. Norvig, J. Orwant, et al. Quantitative analysis of culture using millions of digitized books. *science*, 331(6014):176, 2011.

[69] Rada Mihalcea, Carmen Banea, and Janyce Wiebe. Learning multilingual subjective language via cross-lingual projections. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 976, 2007.

[70] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of NAACL-HLT*, 2013.

[71] A. Mnih and G. Hinton. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, pages 641–648. ACM, 2007.

[72] A. Mnih and G.E. Hinton. A scalable hierarchical distributed language model. *Advances in neural information processing systems*, 21:1081–1088, 2009.

[73] F. Morin and Y. Bengio. Hierarchical probabilistic neural network language model. In *Proceedings of the international workshop on artificial intelligence and statistics*, pages 246–252, 2005.

[74] M.J. Nigrini and L.J. Mittermaier. The use of benford's law as an aid in analytical procedures. *Auditing*, 16:52–67, 1997.

[75] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *LREC*, 2010.

[76] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135, 2008.

[77] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[78] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 701–710, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2956-9. doi: 10.1145/2623330.2623732. URL `http://doi.acm.org/10.1145/2623330.2623732`.

[79] E. Peters, D. Västfjäll, P. Slovic, CK Mertz, K. Mazzocco, and S. Dickert. Numeracy and decision making. *Psychological Science*, 17(5):407, 2006.

[80] Matt Post, Chris Callison-Burch, and Miles Osborne. Constructing parallel corpora for six indian languages via crowdsourcing. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 401–409. Association for Computational Linguistics, 2012.

[81] Bruno Pouliquen, Ralf Steinberger, Camelia Ignat, Irina Temnikova, Anna Widiger, Wajdi Zaghouani, and Jan Zizka. Multilingual person name recognition and transliteration. *arXiv preprint cs/0609051*, 2006.

[82] Delip Rao and Deepak Ravichandran. Semi-supervised polarity lexicon induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 675–682. Association for Computational Linguistics, 2009.

[83] Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA.

[84] Robert Remus, Uwe Quasthoff, and Gerhard Heyer. Sentiws-a publicly available german-language resource for sentiment analysis. In *LREC*, 2010.

[85] Philip Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *arXiv preprint arXiv:1105.5444*, 2011.

[86] Job Schepens, Ton Dijkstra, Franc Grootjen, and Walter JB van Heuven. Cross-language distributions of high frequency and phonetically similar cognates. *PloS one*, 8(5):e63006, 2013.

[87] Michel Simard, George F Foster, and Pierre Isabelle. Using cognates to align sentences in bilingual corpora. In *Proceedings of the 1993 conference of the Centre for Advanced Studies on Collaborative research: distributed computing-Volume 2*, pages 1071–1082. IBM Press, 1993.

[88] Steven Skiena and Charles B Ward. *Who's Bigger?: Where Historical Figures Really Rank*. Cambridge University Press, 2013.

[89] Benjamin Snyder and Regina Barzilay. Unsupervised multilingual learning for morphological segmentation. In *ACL*, pages 737–745. Citeseer, 2008.

[90] Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. Semantic compositionality through recursive matrix-vector spaces.

In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2012.

[91] Spanishcognates.org. Spanish and english cognates, `http://spanishcognates.org`, 2014.

[92] Prayut Suwanvisat and Somboon Prasitjutrakul. Thai-english cross-language transliterated word retrieval using soundex technique. In *Proceesings of the National Computer Science and Engineering Conference, Bangkok, Thailand*, 1998.

[93] Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. Lexicon-based methods for sentiment analysis. *Computational linguistics*, 37(2):267–307, 2011.

[94] J. Turian, L. Ratinov, and Y. Bengio. Word representations: a simple and general method for semi-supervised learning. *Urbana*, 51:61801, 2010.

[95] Brown University. False cognates between spanish and english, `http://www.brown.edu/Departments/LRC/pluma/voc_false_cognates.pdf`, 2014.

[96] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85, 2008.

[97] Walter JB Van Heuven, Ton Dijkstra, and Jonathan Grainger. Orthographic neighborhood effects in bilingual word recognition. *Journal of Memory and Language*, 39(3):458–483, 1998.

[98] Leonid Velikovich, Sasha Blair-Goldensohn, Kerry Hannan, and Ryan McDonald. The viability of web-derived polarity lexicons. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 777–785. Association for Computational Linguistics, 2010.

[99] Paola Virga and Sanjeev Khudanpur. Transliteration of proper names in cross-lingual information retrieval. In *Proceedings of the ACL 2003 workshop on Multilingual and mixed-language named entity recognition-Volume 15*, pages 57–64. Association for Computational Linguistics, 2003.

[100] Derry Tanti Wijaya and Reyyan Yeniterzi. Understanding semantic change of words over centuries. In *Proceedings of the 2011 international workshop on DETecting and Exploiting Cultural diversiTy on the social web*, pages 35–40. ACM, 2011.

[101] Wikipedia.org. List of national capitals in alphabetical order, `http://en.wikipedia.org/wiki/List_of_national_capitals_in_alphabetical_order`, Jan 2014.

[102] Wikipedia.org. Spanish false cognates and false friends with english, `http://en.wiktionary.org/wiki/Appendix:Spanish_false_cognates_and_false_friends_with_English`, 2014.

[103] www.ethnologue.com. Phylogenetic tree of language families, `http://www.ethnologue.com/browse/families`, 2014.

[104] Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Technical Report CMU-CALD-02-107, Carnegie Mellon University, 2002.

[105] Will Y Zou, Richard Socher, Daniel Cer, and Christopher D Manning. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1393–1398, 2013.