A Graph Based Approach to Text Normalization

Abstract

Keywords: Text Normalization, Twitter, micro-blogs, social media

1. Introduction

Social text has become an enormous part of our lives. We are moving towards to an era that we will be talking using machines more than we talk to each other. Social platforms make mass amounts of people communicate via typed or transcribed text. That is the era that the news are spreading digitally via social media other than edited newspaper articles.

With these in mind, it has been also starting of a new era for text analytics researches. The recent studies on social media such that Stock Prediction[1], politeness detection[2], disaster detection[3] tries to lighten up the road of this digitized future of ours.

Therefore analyzing social media text is a challenge for itself. Due to its noisy nature, many NLP tools are performing poorly on social media text[4]. The problems that noise in the social media text generates for NLP tools can be overcome by some preprocessing steps.

Unlike spoken and written language, digitized language has its own form and nature. Since the beginning of World Wide Web, internet has it's own slang. *lol* meaning *laughing out loudly*, *xoxo* meaning *kissing*, *4u* meaning *for you* are the oldest examples of this slang. Everyday new slangs as well

as new words such as iTunes and new abbreviations are coming up. It is a huge, an evolving language that has long gone beyond the reach and control of spellcheckers and slang dictionaries.

ppl	people
havent	haven't
tmr	tomorrow
soooo	so
sooon	soon
raight	right
raight	alright

r	are
mor	more
doin	doing
n	and
friiied	fried
finge	finger
kissin	kissing

Table 1: Example of noisy tokens and their normalized form

Text normalization is a preprocessing step to restore noisy forms of text to its original(canonical) form[5] to make use in NLP applications or more broadly to understand the digitized text better. For example *talk 2 u later* can be normalized as *talk to you later* or similarly *enormoooos, enrmss, enourmos* can be normalized as *enormous*. Those noisy tokens are referred as Out of Vocabulary(OOV) words. Normalization task restores OOV words to their In Vocabulary (IV) form.

However not every OOV word should be considered for normalization. The social text is continuously evolving with new words and named entities that are not in the vocabularies of the systems [6]. The OOV tokens that should be considered for normalization are referred to as ill-formed words. Oppositely an OOV word can sometimes lexically fit an IV word (ex: *tanks* is both an IV word and OOV word with the canonical form *thanks*). The

task of recognizing which tokens are OOV, and which of those are ill-formed are beyond the scope of this paper.

In [7] Choudhury et Al. proposes that the OOV words observed in noisy text can be classified into two groups, unintentional and intentional errors. The unintentional errors are caused by (1) pressing of the wrong key, (2) pressing of a key more than the desired number of times, (3) deletion of a character or (4) inadequate knowledge of spelling. As for the intentional errors, they can be categorized into four categories: character deletion("tlk" for "talk", "msg" for "message", "tomoro" for "tomorrow", "mob" for "mobile"), phonetic substitution ("nite" for "night", "bk" for "back", "u" for "you", "m8" for "mate"), abbreviations ("btw" for "by the way", "kgp" for "Kharagpur") and non-standard usage ("wanna" for "want to", "betta" for "better", "sumfin" for "something", "b/c" for "because").

In this paper we propose a new approach to text normalization. A graph based model, which benefits from both lexical, contextual and grammatical features of social text.

2. Related Work

Early work on text normalization has been highly made use of noisy channel model. The first work that has a significant performance improvement over the previous research was [8]. They proposed a novel noisy channel model for spell checking based on string to string edits. Their error model was depending on probabilistic modeling of sub-string transformations. [9] improved this approach by extending the error model with phonetic similarities over words.

[7] developed a supervised Hidden Markov Model based approach for normalizing SMS Texts. [10] has expanded this model by introducing an unsupervised noisy channel model. Rather than using one generic model for all word formations as in [7], they used a mixture model in which each different word formation type is modeled explicitly. [11] proposed a phrase-based statistical MT model for the task. They defined the problem as translating the SMS language to English language.

What these methods are missing is they do not consider contextual features, and observe each token has a unique normalization. However that is not the case for the normalization task. The OOV tokens are ambiguous and without contextual information it is not possible to build models that can disambiguate transformations correctly.

3. Methodology

In this paper, we propose a graph based approach that models both contextual and lexical similarity features among an OOV word and candidate IV words. A high level overview of our system is shown in Figure 1. An input text is first preprocessed by tokenizing and Part-Of-Speech (POS) tagging. If the text contains an OOV word, the normalization candidates are chosen by making use of the contextual features which are extracted from a pre-generated directed word-relatedness graph, as well as lexical similarity features. Lexical similarity features are based on edit distance, longest common subsequence ratio, and double metaphone distance. In addition, a slang dictionary is used as an external resource to enrich the normalization candidate set. The details of the approach are explained in the following

sub-sections.

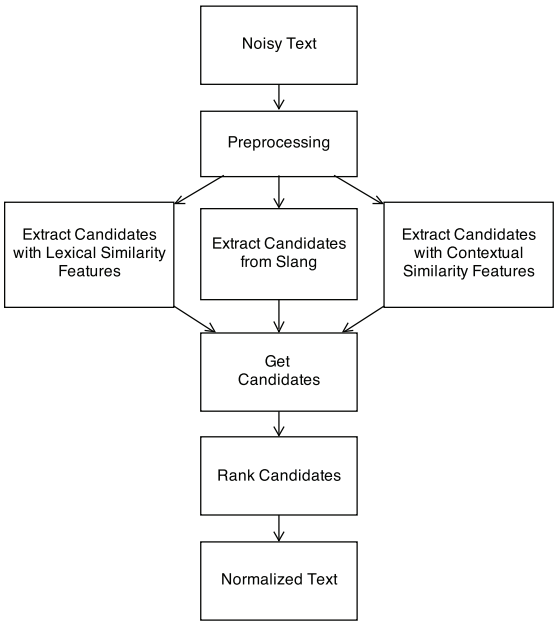

Figure 1: High level overview of our system

3.1. Preprocessing

Tokenization is the first step in our system. Tokenization is the process of breaking the text into words, numbers, symbols, emoticons in other words the smallest meaningful elements of the text called tokens. After tokenization, next in the pipeline is POS tagging each token using a social media POS tagger. Unlike the normal POS taggers, social media POS taggers [12][13] provide a broader set of tags that is special to the social text. By this extended set of tags we can identify tokens such as discourse markers (rt for retweets, cont. for a tweet whose content follows up in the coming tweet) or

URLs. So that we can process those tokens within their context.

As shown in Table 2, after preprocessing, each token is assigned a POS tag with a confidence measure between 0 and 1. Later, we use these confidence scores in calculating the weight of edges in our context graph.

Token	POS tag	Accuracy
with	P	0.9963
a	D	0.998
beautiful	A	0.9971
smile	G	0.9712

Token	POS tag	Accuracy
w	P	0.7486
a	D	0.9920
beatiful	A	0.9733
smile	N	0.9806

Table 2: POS tagger output of samples

3.2. Graph construction

The graph(See Figure 2) is build using a big dataset of social media text. After preprocessing, we traverse each entry in the dataset and extract nodes and edges.

We define a node with four properties *id, oov, freq, tag*. The token itself plus it's POS tag forms the *id* field. *freq* property indicates the node's frequency count in the dataset. *oov* field is set to True if the token is a OOV word. Following Han et al. we used GNU Aspell dictionary (v0.60.6) to determine whether a word is OOV or not.

In the word-relatedness graph, each node is a unique set of a token and a POS tag (see Table 3). This helps us to identify the tokens not only lexically and contextually but also (in terms of POS tags) grammatically.

For example if the token *smile* has been seen frequently as a Noun and

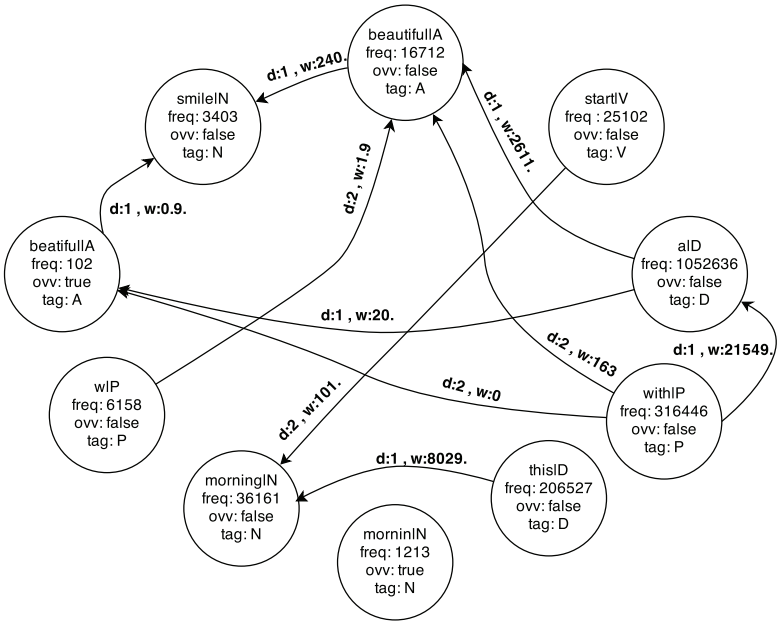

Figure 2: Sample edges and sample nodes from the word-relatedness graph

a Verb and not in other forms in the dataset (Ex: Table4), this means that it is not a good candidate for a Pronoun OOV token (that is a Pronoun). Conversely, if *smile* is lexically and contextually similar enough, it is a good candidate for a Noun or Verb OOV token.

Edges are built depending on the relatedness metrics. One can classify two nodes as related if they satisfy both of the following rules:

- Two nodes are conceptually related if they co-occurs within a word distance of d_t in an entry.
- Each node should have a minimum frequency of f_t in the whole dataset.

The edges follow the flow in the entries thus have a direction from the

Let's$_L$ start$_V$ this$_D$ morning$_N$ w$_P$ a$_D$ beatiful$_A$ smile$_N$.$_,$

Tokens	Let's, start, this, morning, w, a, beatiful, smile, .
Nodes	Let's|L, start|V, this|D, morning|N, w|P, a|D, beatiful|A, smile|V, .|,
Edges	{Let's|L, start|V , distance:1},{Let's|L, this|D, distance:2}, ... {a|D, beatiful|A, distance:1}, {a|D, smile|V, distance:2}, {beatiful|A, smile|V, distance:1}

Table 3: Sample sentence with POS tags, Tokens, The Word-relatedness Graph Nodes and Edges

node id : smile|A , freq : 3, oov : False, tag : A
node id : smile|N , freq : 3403, oov : False, tag : N
node id : smile|V , freq : 2796, oov : False, tag : V

Table 4: Example nodes including token smile with a frequency greater than 0

earlier seen token to the coming token. For example the edges in Table 5 would be derived from a text including the phrase "with a beautiful smile". The *from* property indicates the first word and *to* is the latter in the phrase. Direction and the distance together hold a unique triplet. For each two node with a specific distance there is an edge with a positive weight if that two nodes are related. Each co-occurrence of two related nodes increases the weight of the representing edge with an average of nodes' POS tag confidence score in that specific entry. If we are to expand the graph with our example phrase using the given POS tags and accuracies from Table 2, the increase

in the weights would be respectively $0.9963 + 0.9712/2$, $0.998 + 0.9712/2$ and $0.9971 + 0.9712/2$.

from : with|P, to : smile|N, dis : 3, weight : 72.24415
from : a|D, to : smile|N, dis : 2, weight : 274.37365
from : beautiful|A, to : smile|N, dis : 1, weight : 240.716

Table 5: Example edges from sample phrase "with a beautiful smile"

3.3. Graph Based Contextual Similarity

When we have an entry to normalize, next step after preprocessing is finding normalization candidates for each OOV token in the entry. For each ill-formed OOV token t_i in a given entry, we first list the nodes that is related to t_i. The list includes all the nodes related to t_i and their positional distance to the t_i in the entry. We will refer to this list as neighbour list NL_i. In Table 6 you can find a sample neighbour list for the OOV token beatiful|A from the sample sentence in Table 3.

w|P, position: -2
a|D, position: -1
smile|V, position: 1

Table 6: Example neighbour list for the OOV node beatiful|A

For each neighbour node n_{ij} in the NL_i we traverse the graph and find the edges from or to the node (n_{ij}). The resulting edge list EL_{ij} has edges in the form of (n_{ij}, candidate) or (candidate,n_{ij}) and includes all the nodes that are related to the node n_{ij}. Here the neighbour node can be an OOV node but

the candidate node chosen among the IV nodes. We filter the edges in EL_{ij} by the relative distance of n_{ij} to the t_i given in the NL_i. Each (neighbour, candidate) tuple should have the exact distance as the neighbour and OOV token has.

One last step is to conduct a POS tag filtering on the edges in EL_{ij}. Each candidate node should have the same POS tag with its OOV token. For the OOV token t_i which has tag T_i, all the edges that include candidates with a tag other than T_i are removed from the edge list EL_{ij}. Thus EL_i contains edges only with c_{ik}s that is tagged with T_i.

Each edge in the EL_{ij} has a neighbour node n_{ij}, a candidate node c_{ik} and an edge weight ew_{ijk}. The edge weight, represents the likelihood or the degree of the relatedness between the the neighbour node n_{ij} and candidate node c_{ik}(Eq 1). Since we are looking for candidate nodes that is most likely to be the correct canonical form of the OOV word t_i, this metric is a good indicator. However, weight of the common phrases are much more higher than the average weight. That results in favouring the words with higher frequencies like stop words. To avoid this we normalize the edge weight ew_{ijk} with the frequency of the neighbour node n_{ij}(See Eq 2).

$$ew(n, pos, c) = \begin{cases} w : (n, c, distance = |pos|, weight = w), & \text{if } pos < 0 \\ w : (c, n, distance = |pos|, weight = w), & \text{otherwise} \end{cases} \quad (1)$$

$$ewNormalized(n, pos, c) = ew(n, pos, c)/frequency(c) \quad (2)$$

We have a metric that assures contextual similarity based on binary relatedness. However we need more than binary relatedness to achive a compre-

hensive contextual coverage. To assure this broader coverage, one contextual similarity feature is built based on sum of the binary relatedness scores of several neighbours. For a candidate node c_{ik} the total edge weight score is the sum of ew_{ijk} which is the edge weights coming from different neighbours of the OOV token t_i. You can find the formula in Equation 4. We expect this contextual similarity feature to favour and XXX(find) the candidates which are (1)related to as many neighbours as possible and (2)have a high relatedness score with each neighbours.

$$edgeWeightScoreNeigh(t, n, c, pos) = \sum_{n,c \in EL(t,n)} ewNormalized(n, pos, c) \tag{3}$$

$$edgeWeightScore(t, c) = \sum_{n,pos \in NL(t)} contSimCostNeigh(t, n, c, pos) \tag{4}$$

Since the graph includes both OOV and IV tokens and our OOV detection depends on the spellchecker which excepts some OOV tokens that has the same spelling with another IV word as IV, to be able to propose better canonical forms, the frequency of normalization candidates has been also added to the contextual similarity feature. Nodes with higher frequencies lead to tokens', that are in their most likely grammatical forms.

To calculate the final contextual similarity cost, we sum the total edge weight score and a frequency score between 0 and 1 (proportional to their frequency) (See Eq 5). Since the total edge weight score is our primary contextual resource, while calculating the total contextual similarity metric, we assigned to the frequency feature, only half weight of the total edge weight score.

$$contScore(t,c) = \lambda_a edgeWeightScore(t,c) + \frac{\lambda_a}{2} freqScore(c) \quad (5)$$

Hereby, we have the candidate list CL_i for the OOV token t_i that includes all the unique candidates in EL_i and their contextual similarity costs calculated.

3.4. Lexical Similarity

Following [5],[6], we built our lexical similarity features over standard edit distance [14], double metaphone(phonetic edit distance) [15] and longest common subsequence ratio over edit distance(LCRS) [16].

We calculated both edit distance, phonetic edit distance and LCSR of each candidate in CL_{ij} and OOV token t_i.

We used edit distance and phonetic edit distance to filter the candidates. Any candidate in CL_{ij} with an edit distance greater than ed_t and phonetic edit distance greater than ped_t to t_i has been removed from the candidate list CL_{ij}.

For the rest of the candidates we calculated the total lexical similarity score(Eq 6) using LCSR and edit distance score [1]. Since the main lexical feature is LCSR, we applied the same distributions we used in contextual similarity score in calculating lexical score too.

$$lexScore(t,c) = \lambda_a LCSR(t,c) + \frac{\lambda_a}{2} editDistScore(t,c) \quad (6)$$

[1] an approximate string comparator measure (between 0.0 and 1.0) using the edit distance `https://sourceforge.net/projects/febrl/`

Additionally, sistemin ksa entry'ler(context balants kurulamayacak kadar ksa), ve context'ten bamsz token'lar da kapsayabilmesi iin graph' tekrar dolap contextual similarity feature' tamasa da yukardaki filtreye uyuyorsa aday listesine ekledik.

son olarak slang dictionaryden faydalanarak aday yelpazemizi genilettik.

3.5. Ranking

butun parametetreler ayn.Hassan'larn lambdas

4. Experiments

4.1. Data sets

We used a large amount of social media text to construct our co-occurrence graph. We extracted 15GB tweets from Stanford's 476 million Twitter Dataset[17]. After tokenization we removed tokens POS tagged as mention(@brendon), discourse marker (ex: RT), URL or email addresses, emoticons, numerals and punctuations, we used remaining tokens to build the graph.

For tokenization and POS tagging the tweets we used CMU Ark Tweet Tagger[12][13]. Ark Tweet Tagger is a social media specific tagger and reported to perform 95% accuracy over social media text.

The POS tagset of ark tagger includes some extra tags besides the standard part of speech tags that is specific to social media: Urls and emoticons; Twitter hastags # ; twitter at-mentions (). One other tag that is special to social media is m̃eans the token is specific to a discourse function of twitters. Lastly G stands for miscellanneous words including multiword abbreviations like btw(by the way), nw(no way), smh(somehow).

After constructing the graph we only kept the nodes with a frequency greater than 8 and the edges with weight greater than 1. We had remaining 105428 nodes and 46609603 edges in the graph.

Han'larn data setini eveluation icin kullandm.

5. Results

6. Conclusion

[1] J. Si, A. Mukherjee, B. Liu, Q. Li, H. Li, X. Deng, Exploiting topic based twitter sentiment for stock prediction, in: ACL (2), The Association for Computer Linguistics, 2013, pp. 24–29.

[2] C. Danescu-Niculescu-Mizil, M. Sudhof, D. Jurafsky, J. Leskovec, C. Potts, A computational approach to politeness with application to social factors, in: [18], 2013, pp. 250–259.

[3] T. Sakaki, M. Okazaki, Y. Matsuo, Earthquake shakes twitter users: Real-time event detection by social sensors, in: Proceedings of the 19th International Conference on World Wide Web, WWW '10, ACM, New York, NY, USA, 2010, pp. 851–860. URL: http://doi.acm.org/10.1145/1772690.1772777. doi:10.1145/1772690.1772777.

[4] A. Ritter, C. Cherry, B. Dolan, Unsupervised modeling of twitter conversations, in: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Association for Computational Linguistics, 2010, pp. 172–180. URL: \url{http://scholar.google.de/scholar.bib?

q=info:5hTIjtmFJKAJ:scholar.google.com/&output=citation&hl=de&as_sdt=0&ct=citation&cd=67}.

[5] B. Han, T. Baldwin, Lexical normalisation of short text messages: Makn sens a #twitter, in: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1, HLT '11, Association for Computational Linguistics, Stroudsburg, PA, USA, 2011, pp. 368–378. URL: http://dl.acm.org/citation.cfm?id=2002472.2002520.

[6] H. Hassan, A. Menezes, Social text normalization using contextual graph random walks, in: [18], 2013, pp. 1577–1586.

[7] M. Choudhury, R. Saraf, V. Jain, A. Mukherjee, S. Sarkar, A. Basu, Investigation and modeling of the structure of texting language, Int. J. Doc. Anal. Recognit. 10 (2007) 157–174.

[8] E. Brill, R. C. Moore, An improved error model for noisy channel spelling correction, in: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics, ACL '00, Association for Computational Linguistics, Stroudsburg, PA, USA, 2000, pp. 286–293. URL: http://dx.doi.org/10.3115/1075218.1075255. doi:10.3115/1075218.1075255.

[9] K. Toutanova, R. C. Moore, Pronunciation modeling for improved spelling correction, in: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL '02, Association for Computational Linguistics, Stroudsburg, PA, USA, 2002, pp. 144–151.

URL: http://dx.doi.org/10.3115/1073083.1073109. doi:10.3115/1073083.1073109.

[10] P. Cook, S. Stevenson, An unsupervised model for text message normalization, in: Proceedings of the Workshop on Computational Approaches to Linguistic Creativity, CALC '09, Association for Computational Linguistics, Stroudsburg, PA, USA, 2009, pp. 71–78. URL: http://dl.acm.org/citation.cfm?id=1642011.1642021.

[11] A. Aw, M. Zhang, J. Xiao, J. Su, A phrase-based statistical model for sms text normalization, in: Proceedings of the COLING/ACL on Main Conference Poster Sessions, COLING-ACL '06, Association for Computational Linguistics, Stroudsburg, PA, USA, 2006, pp. 33–40. URL: http://dl.acm.org/citation.cfm?id=1273073.1273078.

[12] O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, N. Schneider, N. A. Smith, Improved part-of-speech tagging for online conversational text with word clusters, in: HLT-NAACL, The Association for Computational Linguistics, 2013, pp. 380–390.

[13] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, N. A. Smith, Part-of-speech tagging for twitter: Annotation, features, and experiments, in: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2, HLT '11, Association for Computational Linguistics, Stroudsburg, PA, USA, 2011, pp. 42–47. URL: http://dl.acm.org/citation.cfm?id=2002736.2002747.

[14] V. Levenshtein, Binary Codes Capable of Correcting Deletions, Insertions and Reversals, Soviet Physics Doklady 10 (1966) 707.

[15] L. Philips, The double metaphone search algorithm, C/C++ Users J. 18 (2000) 38–43.

[16] D. Contractor, T. A. Faruquie, L. V. Subramaniam, Unsupervised cleansing of noisy text, in: Proceedings of the 23rd International Conference on Computational Linguistics: Posters, COLING '10, Association for Computational Linguistics, Stroudsburg, PA, USA, 2010, pp. 189–196. URL: http://dl.acm.org/citation.cfm?id=1944566.1944588.

[17] J. Yang, J. Leskovec, Patterns of temporal variation in online media, in: I. King, W. Nejdl, H. Li (Eds.), WSDM, ACM, 2011, pp. 177–186.

[18] conf/acl/2013-1, Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers, The Association for Computer Linguistics, 2013.