Toward Inferring the Age of Twitter Users with their Use of Nonstandard Abbreviations and Lexicon

Nathaniel Moseley
Department of Computer Science
Rochester Institute of Technology
Rochester, NY 14623
nmoseley@csh.rit.edu

Cecilia Ovesdotter Alm
Department of English
Rochester Institute of Technology
Rochester, NY 14623
coagla@rit.edu

Manjeet Rege
Graduate Programs in Software
University of St. Thomas
St. Paul, MN 55105
rege@stthomas.edu

***Abstract*—Automatically determining demographic profile attributes of writers with high accuracy, based on their texts, can be useful for a range of application domains, including smart ad placement, security, the discovery of predator behaviors, enabling automatic enhancement of participants profiles for extended analysis, and various other applications. Attributes such as author gender can be determined with some amount of success from many sources, using various methods, such as analysis of shallow linguistic patterns or topic. Author age is more difficult to determine, but previous research has been somewhat successful at classifying age as a binary (e.g. over or under 30), ternary, or even as a continuous variable using various techniques. In this work, we show that word and phrase abbreviation patterns can be used toward determining user age using novel binning. Notable results include classification accuracy of up to 82.8%, which was 67.0% above relative majority class baseline when classifying user ages into 10 equally sized age bins using a support vector machine classifier and PCA extracted features (including n-grams) and 50.8% (33.7% above baseline) when using only abbreviation features. Also presented is an analysis of the evident change in abbreviation use over time on Twitter.**

I. Background

In the area of computational linguistics, there is an interest in determining latent attributes of an author. Many texts are now available online or are created online. One such online corpora, increasingly receiving attention, are *microblogs*; these are texts, such as on Twitter or Facebook, which are comprised of short, often character-limited messages. Due to their temporally sequenced and pervasive nature, these texts are an interesting area for research. However, as users of these services are not expected to write using orthographically standard language, and also must conform to character restrictions (up to 140 characters in the case of Twitter), the texts are often noisy and could present challenges to work with. Users often abbreviate words and phrases in non-standard ways in order to convey their messages in fewer characters [1].

A variety of work has been published that focuses on linguistic analysis for author age in on- and offline texts, much of which focuses on lexical and contextual clues, such as analyzing topic and genre or n-gram patterns. Rosenthal and McKeown analyzed online behavior associated with blogs (*i.e.* usually more comprehensive contents than tweets) and found that behavior (number of friends, posts, time of posts, *etc.*) could effectively be used in binary age classifiers, in addition to linguistic analysis techniques similar to those mentioned above [2].

With respect to examining another demographic feature, Sarawgi *et al.* explored non-contextual syntactic patterns and morphological patterns to find if gender differences extended further than topic analysis and word usage could indicate. They used probabilistic context-free grammars, token-based statistical language models, and character-level language models, that learn morphological patterns on short text spans. With these, they found that gender is evident in patterns at the character-level, even in modern scientific papers [3].

Much earlier computational and linguistic analysis focused on formal writing or conversation transcripts, which generally conform to standard English corpora and dialects, syntax, and orthography. Today, in many areas, there is a strong interest in leveraging information and user information available in online texts which do not tend toward prescriptive standards, including SMS messages and social networking blurbs.

Some research involves adapting existing linguistic tools to handle nonstandard lexicons, such as those found on Twitter. Gimpel *et al.* developed a part-of-speech tagger by introducing new types of text such as emoticons and special abbreviations used on Twitter, in addition to traditional parts of speech [4]. Part-of-speech analysis can subsequently be used to aid normalization of noisy text, or the part-of-speech patterns themselves can be used as classification features.

Much of the time, noisy texts, such as those on Twitter, must be cleaned or normalized before performing traditional text analysis. Kaufmann and Kalita developed such a normalization system. They applied pre-processing techniques to normalize orthographic modifications as well as twitter-specific elements (@-usernames and # hashtags). They found that after limited pre-processing, a machine translation approach, translating the noisy Twitter texts into standard English, worked well [5].

Gouws *et al.* expanded the approach of Contractor *et al.* [6] and used the pre-processing techniques of Kaufmann and Kalita [5] in order to identify types of creative lexical transformations resulting in OOV tokens in Twitter messages. Gouws *et al.* showed that these lexical transformations could be used to differentiate between user geographical regions (as determined by time zones) and client used to post tweets. These transformations are discussed further in section III.

Additionally, Wagner explains writing and speech patterns change over time as a person learns a language and develops socially; she calls this *age grading* [7]. Language use changes

IEEE IRI 2014, August 13-15, 2014, San Francisco, California, USA
978-1-4799-5880-1/14/$31.00 ©2014 IEEE

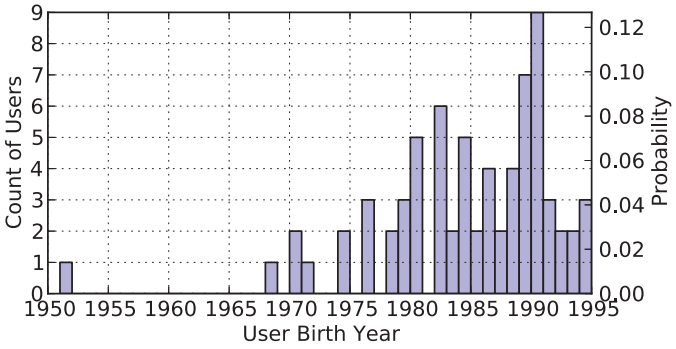


Figure 1: Distribution of participant birth years. Due to the age restrictions of the Twitter service, as well as participation solicitation methods, there is little or no data for certain population groups, *e.g.* children and older adults.

as an individual grows through life stages, such as childhood, adolescence, and adulthood. Language is acquired in childhood and linguistic interaction skills are developed. Through adolescence, individual social and linguistic identity is explored and language use continues to develop through adulthood. Sometimes, these changes are made along with community language use change; Wagner refers to this as *generational change*.

In sum, prior work suggests that text-based age prediction is tenable but also leaves plenty of room for additional study of the text-based age prediction problem. Lastly, as language changes over time, it is reasonable to expect language use changes to be observable in Twitter posts over time. In this paper, we explore if lexical choices and the abbreviation patterns defined by Gouws *et al.* can help identify user age on Twitter, as they are indicative of a user's region and Twitter client [8]. We present our formative assessment of findings regarding user age classification using word and phrase abbreviations found in Twitter messages using a new Twitter data set, annotated by Twitter users themselves and made openly available for research. Additionally, we discuss empirically-based observations regarding users' change of linguistic abbreviation features over time in their written Twitter messages through analysis of individual users' lexical transformation use compared to length of time using Twitter.

II. Data Set

Because age data is not directly associated with Twitter accounts, a new data set was developed to enable this study. Twitter usernames and birth years were solicited from a variety of sources. Users were allowed to self label their demographic data through a web form resulting in the collection of over a hundred thousand tweets.

Distribution of user age in the collected data set is shown in Figure 1. It is difficult to do comparisons with much published work with regards to age, but this data set is more influenced by younger and more educated Twitter users when compared to some other Twitter corpus descriptions [1], [9]. Other demographic information in the data set, such as gender and birth month, is somewhat consistent with published research on Twitter demographics. This is similar to the data reflected by Beevolve, which collected user ages when they were defined in a user's profile on Twitter textually. Beevolve found, of those users that self-report their age on their Twitter profile (about 0.45%), almost 90% are under 35. Of those, 73.3% are aged 15-25 [10]. This suggests that younger users are much more willing to divulge their age, which could account for the prevalence of young participants in this data set, as revealing age was mandatory for participation. These biases are part of what makes age prediction such a difficult problem. The data were comprised of a bit over a hundred thousand instances (one per tweet), which were combined in several ways for analysis.

Table I: Feature type names, short descriptions, and examples as found in the data set.

Feature	Example
Single Character Replace a word with a single character, often a phonemic substitution.	("too" → "2")
Word End Drop all but a substring at the end of a word.	("robot" → "bot")
Drop Vowels Drop one or more orthographic vowels.	("seriously" → "srsly")
Word Begin Drop all but a substring at the start of a word.	("though" → "tho")
You to U Replace "you" with "u".	("your" → "ur")
Drop Last Char Drop the last character of a word.	("goin" → "going")
Repeat Letter Repeat letters (often vowels) for emphasis.	("good" → "goooood")
Contraction Standard or invented contraction, often using an aphostrophe.	("them" → "'em")
Th to D Replace a "th" sound with a "d".	("that" → "dat")

The discussion of experimental work below reports on results for the grouping of 75 and 100 tweets per instance. In other words, 75 or 100 tweets from the same user in chronological order were grouped and treated as one instance, then processed for feature vectors. Extra tweets that did not fill a full group were ignored, so that each instance was comprised of a full 75 or 100 tweets. Given the overall collected data set, this resulted in approximately 1000 instances in the case of 100 tweets per instance. Experiments were also conducted that considered groups of 1, 25, 50, and over 100 tweets per instance but they resulted in lower performance; and have been excluded from the below discussion.

III. Features

In order to develop a model for predicting Twitter user age, the data set's associated tweet texts were downloaded

```
Data: tweet text
Result: normalized tweet text
begin generate sentence candidates
    Remove emoticons and HTML artifacts
    tokens ← Tokenize sentence using NLTK + customization
    candidates ← foreach token in tokens do
    Generate substitution candidates and probabilities
        if OOV_but_valid(token) then
            return token, 1.0
        end
        return list of substitution candidates and probabilities for token
    end
    lattice ← generate confusion network for candidates
    replacements ← generate lowest word error sentence from lattice
    return replacements
end
```

Figure 2: Text cleanser algorithm provided by Gouws *et al.*. This work added some customization in tokenization and small improvements.

Table II: Some abbreviations and their generated replacements. In most cases of identified abbreviation types, a suitable replacement was generated, while in others a similar but contextually incorrect replacement was generated (italicized). However, a large number of abbreviations were unidentified because an unrelated replacement was generated, usually when a word was identified as OOV when it should not have been. Also, multi-word acronyms were often considered as OOV, but assigned an incorrect abbreviation type, adding to the number of anomalous abbreviations.

Abbreviation	Replacement
ppl	people
gian	giant
sched	schedule
yr	year
hlep	help
SATs	*seats*
IPO	*ipod*
IDE	*decide*
app	*apple*
tater	*theater*
YouTube	*your*

and then analyzed for abbreviation features. The abbreviation features used are those found by Gouws *et al.* to be most frequent in an overall Twitter corpus [8]. Those features are described in Table I. Based on research showing the predictive capability of behavioral elements of text with respect to age, as well as morphological and lexical patterns with respect to demographic information such as gender, it made sense that such abbreviation features might be indicative of user age, as they were with user time zones and Twitter clients.

In order to label tweets with abbreviation features, collected tweets were fed through the cleanser framework developed by Gouws *et al.*, which attempted to text-normalize each tweet into a standard English sentence. Different stages of the text normalization utilized functions from the python Natural Language Toolkit (NLTK) framework [11] and the SRI Language Modeling Toolkit (SRILM) [12]. The algorithm is outlined in Figure 2, the steps of which are summarized here:
(1) For each tweet, remove any obvious emoticons, HTML artifacts, and punctuation tokens.
(2) Tokenize each tweet into individual word tokens and punctuation using the NLTK *tokenize.punkt* library.
(3) Generate substitution candidates and probabilities for each OOV token using a string subsequence kernel [13].
(4) A word mesh is generated from the list of candidates and probabilities, which is fed into the *lattice-tool* program of SRILM to decode into a most likely cleaned sentence, consisting of the candidates with the lowest expected word error.
(5) OOV tokens and determined replacements are recorded for abbreviation analysis.

Two examples of this processing are shown in Figure 3.

Lexical n-gram features were assigned to each tweet based on the tokenization from part (2) above. The n-gram features contained the original text of the tweets before normalization with some processing done by the punkt library. Punctuation was separated from adjacent word tokens and treated as its own token, except for contractions, such as *don't*, which were split into two base words (*do* and *n't*).

The abbreviation features are determined on a per-tweet level, based on a per-token analysis using the algorithm outlined in Figure 4. Each OOV token and replacement pair are analyzed with regex patterns for abbreviation types. A percentage vector is assigned to each instance, which reflects the percentage of tokens in the instance which conform to each abbreviation type. In order to investigate effects of data sparsity on results, the percentages are further generalized to a boolean vector, which describes if a given abbreviation feature type was used at all in an instance. In the experimental sections, these vectors are referred to as *percentage-abbreviation* and *boolean-abbreviation* features.

While Gouws reported 90% coverage with the 9 defined abbreviation types with a large Twitter data set [8], those types only cover 43% of the found abbreviation patterns in this data set. A contributing factor in this difference is likely that the data set primarily captures more educated persons than other studies have found in a general Twitter populous [14]. Many collected tweets are written in mostly standard English with standard English syntax. Additionally, the LA Times corpus and Han and Baldwin's tweet corpus used to train the sentence normalizer used in this study may not include newer invented words or proper nouns. As such, many tokens are replaced with unnecessary substitutions, such as shown in Table II, creating anomalous abbreviation patterns that do not fit into the defined categories at a higher rate than an older Twitter corpus or one representing different demographics.

About 30 thousand of the gathered tweets (30% of a bit over a hundred thousand) did not have any recorded word and phrase abbreviation features, as they were written in standard English. For this reason, each experiment was run on an

Wehn	yuo	cnnaot	raed,	noe	hrund	ftory	ccrhaetars	mean	noinhtg.	...
↓	↓	×	↓	◯	↓	×	↓	↓	↓	
when	you	cinnamon	read,	noe	hundred	factory	characters	mean	nothing.	...
Dude	thx	shes	a	gr8	grl.	...				
	↓	×		↓	↓					
Dude	thanks	she	a	great	girl.	...				
Siri's	like	an	advanced	IRC	channel	bot.	...			
× ↓				×		×				
spirit is	like	an	advanced	rich	channel	both.	...			

Figure 3: Three tweets identified as having abbreviations and their generated normalized equivalents. The normalizer works quite well on some tweets, such as with phonemic substitutions or misspellings, but on others it can perform quite poorly, such as with newer vocabulary. ↓ represents correct normalization, × represents an incorrect normalization, and ◯ represents a word that should have been changed, but was not, according to human annotation.

Table III: Age values covered by equal size classification bins. Bin time ranges were generated so that the number of instances were as equal as possible between bins. Instances are assigned a class based on the age of the user (in years) at the time of writing the tweet or tweets represented by the instance's feature data.

Bins	Age Range
2	$<= 25, <= 61$
4	$<= 22, <= 25, <= 30, <= 61$
6	$<= 20, <= 22, <= 25, <= 28,$ $<= 32, <= 61$
8	$<= 20, <= 22, <= 23, <= 25,$ $<= 28, <= 30, <= 33, <= 61$
10	$<= 19, <= 21, <= 22, <= 23, <= 25,$ $<= 27, <= 28, <= 30, <= 33, <= 61$

instance set with all tweets, as well as one filtered to only include tweets which exhibited at least one abbreviation feature type. This division was done on a per-tweet level, so when several tweets were grouped in one instance, each joined tweet had at least one identified feature. (As one would expect, this filtering did not positively impact results for experiments based just on lexical n-grams.) All noted results were obtained using filtered data for comparison.

Using the age data provided by Twitter users' themselves and the timestamps of users' collected tweets, each tweet instance was assigned an age by subtracting the user's age as a UTC timestamp from the tweet's UTC timestamp. For grouped instances, the assigned age was the average age of all contained tweets' ages. This method was selected to isolate possible changes in a user's language use over time. Each grouping covered a small amount of time, even if the user had been tweeting for years. Using the assigned instance age, each instance was assigned a classification bin. Instances were grouped into 2, 4, 6, 8, and 10 bins to create separate instance sets for initial experiments. The age ranges of the different binning types are shown in Table III. Results reported here are from experiments with 10 bins. Experiments using this binning most frequently outperformed experiments on lower numbers of bins in terms of accuracy improvement over baseline.

The below discussion reports on equal-size age bins, in which each bin covered roughly the same number of instances. Equal-width bins performed worse, which one would expect since it leads to a class imbalance, as well as empty bins for some bin counts. Equal-size binning could avoid such problems caused by ages in the data set being unevenly distributed. However, with a large number *k*, *k* bins can lead to bins being created with less useful spans, such as a year or less. Additionally, although single-tweet per instance grouping in early experimentation showed fairly even spread in equal-size bins, grouping tweets into instances of many tweets, thus resulting in fewer instances, affected bin distribution. In grouped instances, majority class baselines were as much as 6% greater than would be expected with a perfectly even split. This could partially account for experiments on 10 bins often outperforming those on 6 or 8 bins. Future work could experiment with different ways to bin age, perhaps using optimization to find the best binning approach, or use different forms of data sampling, such as treating age as a continuous variable, as demonstrated by Nguyen *et al.* [15].

Table IV: Some n-grams generated and their rate of occurrence as a percentage of all collected tweets. Very few trigrams were selected, as their frequency was very low. Of those that were selected, most include some form of punctuation, such as *, too .* shown above.

N-gram	Frequency
thank	0.335
thank you	0.258
, too	0.187
, too .	0.115
ur	0.103

IV. Experiments

The experiments were run using combinations of percentage-abbreviation, boolean-abbreviation, and lexical n-gram feature vectors. The most frequent n-grams ($n = \{1, 2, 3\}$) were selected by pruning a dictionary of n-grams, removing the least frequent n-grams and keeping a set minimum number of n-grams, resulting in about 1200 text features. In experiments using n-grams, feature selection or extraction was run to reduce the number of features, often to around 400 or less. Some selected n-grams are shown in Table IV. These

```
Data: tokenized tweet array
Result: abbreviation feature vectors (percents and
        booleans)
begin
    counts ← vector(0,10)
    foreach pair in tweet array do
        tok ← pair[0]
        rep ← pair[1]
        switch tok do get abbr. for pair
            case rep.replc("you", "u")
                typ ← "you to u"
            case rep.replc("aeiou", "")
                typ ← "drop vowels"
            case rep.replc("aeiouy", "")
                typ ← "drop vowels"
            case rep.substr(0, len(rep) - 1)
                typ ← "drop last character"
            case rep = de_repeat(tok)
                typ ← "repeat letter"
            case rep.endsWith(tok)
                typ ← "word end"
            case rep.startsWith(tok)
                typ ← "word begin"
            case rep.replc("th", "d")
                typ ← "th to d"
            case is_contr(tok, rep)
                typ ← "contraction"
            case tok in rep and len(tok) = 1
                typ ← "single char."
            otherwise
                typ ← "unknown"
            end
        endsw
        increment counts for typ
    end
    percents ← counts/length
    booleans ← counts > 0
end
```

Figure 4: Abbreviation feature assignment algorithm. Some more specific features are subsets of other features. For example, any *drop last character* feature is also a *word begin* feature. The classifications are assigned such that features which are subsets of other features are assigned first (they can not be both due to the subset relationships). Feature testing was done using a combination of regex and recursive logic.

three feature types were used in experiments individually, as well as in three combinations of two types and one of all three types.

Each instance set was run through experiments on all its raw features, as well as features resulting from Weka's best first feature selection algorithm and the Principal Component Analysis (PCA) algorithm for comparison. This was particularly important for experiments with lexical n-grams to keep the feature set a manageable size and ensure classifier run time did not become unfeasible.

Each instance set was run through an experiment (a series of runs of an instance set through a classifier) that utilized each of the set of classifiers below and analyzed using t-tests with a confidence of 0.05 to ensure the results selected for note were statistically better than baseline by more than just random chance.

Table V: SVM classifier results run using only abbreviation features and their combinations. PCA extracted features generally were classified with higher raw accuracy and improvement over baseline (denoted in the Improvement column).

Raw Feature	Accuracy	Improvement
Boolean	21%	5%
Percentage	20%	5%
Bool + Percent	22%	6%
PCA Feature		
Boolean	25%	9%
Percentage	47%	31%
Bool + Perc.	47%	31%

Experiments were run through Weka using a set of basic classifiers and more advanced learning algorithms. The basic classifiers were OneR [16], Naive Bayes [17], and a J48 (C4.5) decision tree [18]. The learning algorithms were a Support Vector Machine (SVM) implementation provided by LibSVM [19], and a Multilayer Perceptron Network (MPN) in Weka. All experiments began with a ZeroR classifier (majority class) as an absolute baseline. These algorithms were selected for comparative purposes. For example, if a basic probability-based classifier performed as well as a longer-running learning algorithm, it might be more useful in further applications to work with basic classifiers.

The train-test data were randomly shuffled and split differently during initial pilot experimentation and later evaluation experimentation to ensure distinct identities of the data splits. Initial experiments for determining instance set and classifier parameters were run with a randomized 80-20 test-train split, averaged over ten runs for the basic classifiers and 3 runs for the more sophisticated learning algorithms. Experiments reported on below were run using 10-fold cross validation, which randomly divides the input instance set into 10 non-overlapping folds. Each classifier-instance set pair was run through cross validation three times and the results were averaged for analysis.

Lastly, longitudinal analysis was run on a per user basis, as described in subsection V-C. The features were analyzed for abbreviation usage versus tweet timestamp to see if abbreviation usage changes could be observed as a user became more experienced with writing on Twitter.

V. Results

A. Abbreviation Feature Experiments

Abbreviation features were found to improve results when compared to a naive baseline. Boolean-abbreviation features, with their added information sparsity, performed least well while still outperforming baselines. Percentage-abbreviation features, however, did quite well, especially when combined with PCA extraction. In most cases, combining boolean-abbreviation and numeric-abbreviation features had little or no

Table VI: SVM classifier results for n-gram feature combinations. In contrast to experiments on just abbreviation features, SVM classification on n-gram based features was frequently outperformed by Multilayer Perceptron Network classification, though not by a large margin. The improvement column shows accuracy improvement over majority class baseline.

Best First Feature	Accuracy	Improvement
N-gram	77%	61%
+ Boolean	72%	56%
+ Percentage	70%	54%
+ Both	70%	54%
PCA Feature		
N-gram	63%	47%
+ Boolean	83%	67%
+ Percentage	83%	67%
+ Both	83%	67%

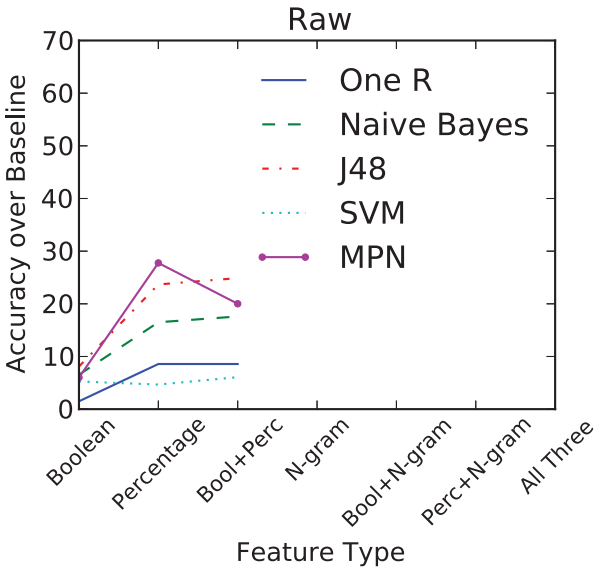


Figure 5: Accuracy improvement of each classifier over baseline with raw feature types, 10 bins, and 75 tweets per instance. Percentage-abbreviation features generally performed better, with a Multilayer Perceptron Network peaking at 28% over majority class baseline. Feature sets that include n-grams are not included in this graph, as feature selection or extraction was required to reduce dimensionality before experimentation, and raw n-grams were not tested.

positive effect, though the J48 classifier did perform a bit better on the combination of the two feature types. Some selected results are shown in Table V, as well as across all feature types and classifiers in Figure 5, Figure 7, and Figure 6 with respect to the majority class baseline. All selected results were from instance sets with 75 tweets per instance and 10 age bins. The best result using abbreviation features alone, however, was from an SVM run on PCA extracted percentage-abbreviation features at 51% accuracy (34% over relative baseline).

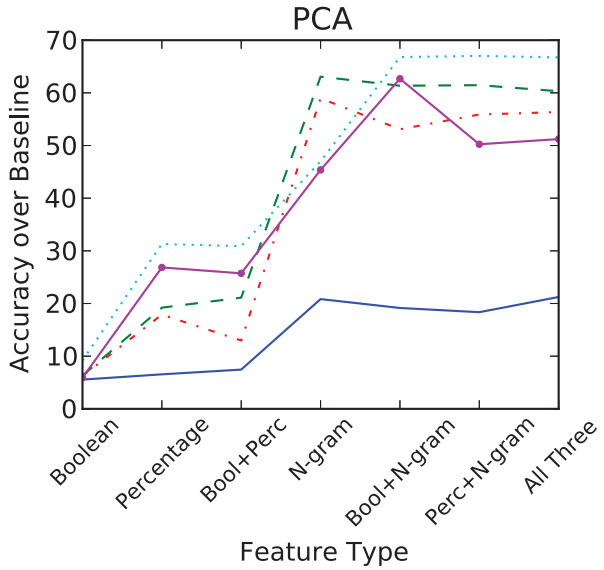


Figure 6: Accuracy improvement of each classifier over baseline with PCA extracted features, 10 bins, and 75 tweets per instance. Lines are defined as in Figure 5. The support vector machine classifier performed better with PCA features than raw or best-first. Additionally, n-gram features did not do as well alone, compared to when combined with abbreviation features. An SVM classifier reached 31% with percentage-abbreviation features, and 67% with n-gram and percentage features.

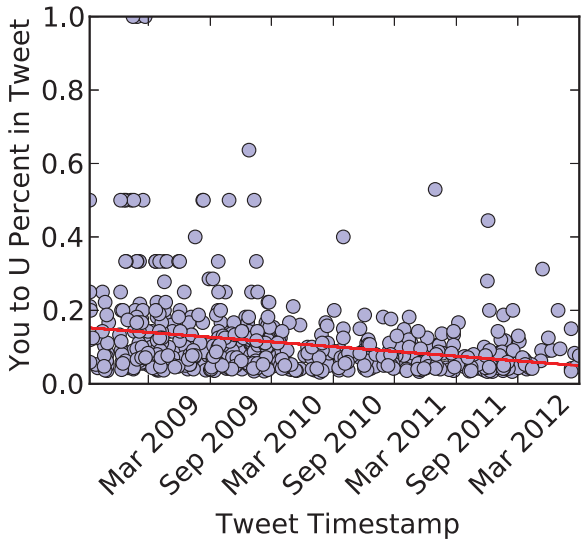
B. *Experiments Including N-grams*

Experiments using solely abbreviation features could not match n-gram features for classification accuracy, as could be expected, except in the case of SVM classification of PCA extracted features. An SVM did not perform as well on purely n-gram features as did other classifiers, but when combined with either type of abbreviation feature, it matched or slightly exceeded the best n-gram results.

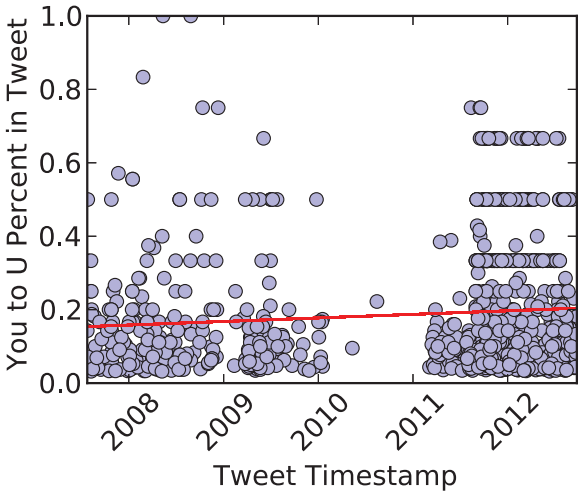
Analysis done on n-grams performs well due to the additional meaning encoded in words and phrases. As can be seen from the SVM results on n-grams and other features, shown in Figure 6 and Figure 7, abbreviation features may be able to improve n-gram analysis results in some cases. N-gram experiment results were best when using best-first selected features, likely due to the large number of features associated with the gathered n-grams. The PCA extraction was able to reduce the feature dimensionality, but, alone, its generated features were not as useful. Some selected results are shown in Table VI.

C. *Longitudinal Analysis*

Longitudinal analysis was performed on the top 10 longest tweeting users in the data set to investigate if a user's abbreviation feature usage changes over time. The top 10 longest tweeting users were selected, because few users in the collected data set provided tweets that covered a large time span. The tweets analyzed covered from 4 to just over 5 years per user. For each selected user, tweets were plotted with UTC Unix epoch timestamp on the x axis (converted to readable dates for visualization) and abbreviation feature percentage on the



(a) The most negative slope at −0.1043, indicating the user tended to use less of the You to U abbreviation feature with more time spent on Twitter (over about four years).



(b) The most positive slope at 0.0501, indicating the user tended to use more of the You to U abbreviation feature with more time spent on Twitter (over about five years).

Figure 8: Plots of *You to U* feature use percentages over time for two users. The most negative and most positively sloped best fit lines are shown here. Each dot represents one tweet, with its timestamp on the X axis and *You to U* percentage use on the Y axis. The slopes correspond to the red best-fit line, normalized by reducing the timestamp (milliseconds since Unix epoch, represented by a *long* integer) to the range 0 to 1, in order to account for differences in users' length of time using the Twitter service. A negative slope corresponds with a decrease in use of an abbreviation feature within all a user's tweets over time, while a positive slope corresponds to an increase in abbreviation use.

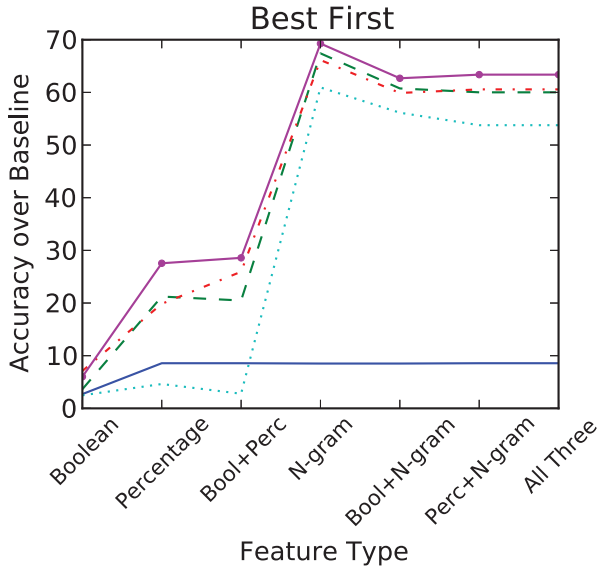


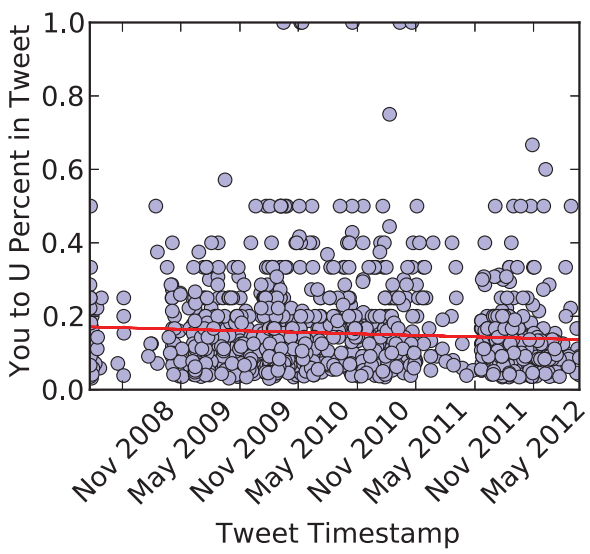
Figure 7: Accuracy improvement of each classifier over baseline with best-first selected features, 10 bins, and 75 tweets per instance. Lines are defined as in Figure 5. Lexical n-gram features performed the best, as could be expected, since n-grams encode additional meaning compared to the abbreviation features. A Multilayer Perceptron Network performed better with best-first selected boolean- and percentage-abbreviation features than raw abbreviation features at 29% above baseline, and with n-grams at 69%.

y axis, creating 9 graphs per user (one per feature type). The collection of points was analyzed with NumPy to obtain a best fit line, which minimizes squared error [20].

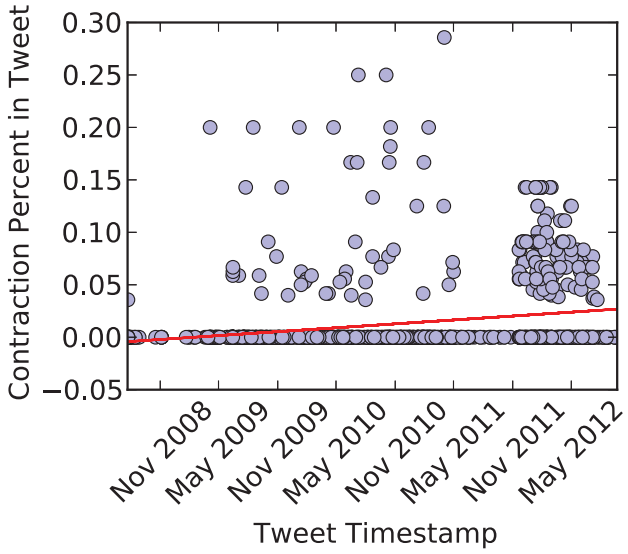
The *You to U* abbreviation type feature had the most noticeable change over time compared to other abbreviation types (*see* Figure 8 and Figure 9). Across all users, calculated best fit slopes were often negative, suggesting that users tended to use less of each abbreviation type as they got more familiar with Twitter.

Sometimes, within one user's analysis (Figure 9), analysis indicated that a user began to use less of one abbreviation type and somewhat more of another. This would suggest that as a user gets more familiar with the Twitter service, they adapt their writing patterns to fit the medium (either its restrictions or changes in prevailing language use) by either writing more standard English as time passes, or adopting a more comfortable abbreviation type, such as contractions (the most common type in this data set), in favor of another.

These changes, such as those common across many users, might also parallel changes in writing patterns with respect to abbreviation usage in other mediums, such as SMS. As smartphones with full keyboards have become more popular, it is easier for SMS users to type full words and utilize sequential SMS messages, which lessens the need for abbreviations. Such changes in SMS messaging may be reflected in other restricted-length texts, such as on Twitter.



(a) A negative slope (−0.0393) for *You to U* feature use with corresponding *Contraction* increase.



(b) A positive slope (0.0352) for *Contraction* feature use with corresponding *You to U* decrease.

Figure 9: Similar opposing abbreviation use changes within one user. In some cases, a positive or negative slope in use of one feature type was joined by one or more other features exhibiting an opposite change at a similar rate. This could suggest that a user may drop one abbreviation type in favor of another as they become more familiar with the Twitter medium. It might also reflect abbreviation use change on Twitter as a whole, which the user was responding to.

VI. Conclusions and Future Work

The experiments showed promise for the use of abbreviation features in age-based classification. Abbreviation classification experiments outperformed relative majority class baselines in the case of percentage-abbreviation features. Boolean-abbreviation features, while interesting to compare, offered much less help when classifying age. These abbreviation features may be a useful in the future as an additional feature for those seeking to perform age and other demographic classification on noisy texts, such as those found on Twitter.We believe that abbreviation feature use should be explored and refined further. It is of particular interest for classifying short messages, such as those found on Twitter, that are linguistically sparse and often abbreviated. Perhaps these features can be used in classifying other user demographics as well, such as gender, ethnicity, etc.

References

[1] B. Han and T. Baldwin, "Lexical normalisation of short text messages: Makn sens a #twitter," in *Proc of HLT*, 2011, pp. 368–378.

[2] S. Rosenthal and K. McKeown, "Age prediction in blogs: A study of style, content, and online behavior in pre- and post-social media generations," in *Proc of ACL - Volume 1*, 2011.

[3] R. Sarawgi, K. Gajulapalli, and Y. Choi, "Gender attribution: Tracing stylometric evidence beyond topic and genre," in *Proc of CoNLL*, 2011.

[4] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and 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*, ser. HLT '11.

[5] M. Kaufmann and J. Kalita, "Syntactic normalization of twitter messages," in *Proc of ICON*, 2010, pp. 149–158.

[6] D. Contractor, T. A. Faruquie, and L. V. Subramaniam, "Unsupervised cleansing of noisy text," in *Proc of COLING*, 2010.

[7] S. E. Wagner, "Age grading in sociolinguistic theory," *Language and Linguistics Compass*, vol. 6, pp. 371–382, June 2012.

[8] S. Gouws, D. Metzler, C. Cai, and E. Hovy, "Contextual bearing on linguistic variation in social media," in *Proc of LSM*, 2011, pp. 20–29.

[9] J. D. Burger, J. Henderson, G. Kim, and G. Zarrella, "Discriminating gender on twitter," in *Proc of EMNLP*, 2011, pp. 1301–1309.

[10] G. Udani. (2012, October) An exhaustive study of twitter users across the world. [Online] Available http://www.beevolve.com/twitter-statistics/. Beevolve Technologies.

[11] E. Loper and S. Bird, "Nltk: the natural language toolkit," in *Proc of ETMTNLP*, 2002.

[12] A. Stolcke, "SRILM - an extensible language modeling toolkit," in *Proc of 7th International Conference on Spoken Language Processing*, 2002.

[13] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, "Text classification using string kernels," *Journal of Machine Learning Research*, vol. 2, pp. 419–444, March 2002.

[14] A. Smith and J. Brenner, "Twitter use 2012," Pew Research Centers Internet & American Life Project, Tech. Rep., 2012.

[15] D. Nguyen, N. A. Smith, and C. P. Rosé, "Author age prediction from text using linear regression," in *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, 2011.

[16] R. C. Holte, "Very simple classification rules perform well on most commonly used datasets." *Machine Learning*, vol. 11, pp. 63–91, 1993.

[17] G. H. John and P. Langley, "Estimating continuous distributions in bayesian classifiers," in *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, 1995, pp. 338–345.

[18] R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann Publishers, 1993.

[19] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 27:1–27:27, May 2011.

[20] M. Turk, "Analysis and visualization of multi-scale astrophysical simulations using python and numpy," in *Proc of 7th Python in Science Conference*, 2008.