Comparative analysis of classifiers identifying politeness markings and application in web-logs

Shagun Jhaver
Dept. of Computer Science
The University of Texas at Dallas
Richardson, Texas 75080
Email: sxj124330@utdallas.edu

***Abstract*—A computational approach to politeness with application to social factors**

Index Terms - Politeness Theory, Classification, SVM

I. Introduction

Politeness, deference and tact have a sociological significance altogether beyond the level of table manners and etiquette books (Goffman 1971:90). Politeness, introduced into linguistics more than forty years ago, has emerged as a vital and rapidly developing area of study. Brown and Levinson's (1978, 1987) classic treatment of linguistic politeness show that politeness strategies are a basis for social order. The concepts inherent to their model have been invoked in much subsequent literature which has focused on linguistic carriers of politeness (e.g., speech acts, syntactic constructions, lexical items, etc.), seeking to quantify them, to compare them across cultures and genders, and to identify universals [1].

Danescu-Niculescu-Mizil, Sudhof, Jurafsky, Leskovec and Potts [2] develop a computational framework for identifying and characterizing the linguistic aspects of politeness. Their investigation is guided by a new corpus of requests annotated for politeness, that they constructed and released. This corpus consists of a large collection of requests from two different sources - Wikipedia and Stack Exchange. Both of these are large online communities in which users frequently make requests of other members.

In this paper, we use this richly labeled data for politeness to construct politeness classifiers using different supervised and unsupervised machine learning algorithms, and present a comparative analysis of the performance of these classifiers. We also study the improvement in classifiers' performance after they use a wide range of lexical, sentiment and dependency features operationalizing key components of politeness theory.

We observe that some of our classifiers achieve near human-level accuracy across different test-sets, which demonstrates the consistent nature of politeness strategies, and we use these classifiers with new data for further analysis of the relation of politeness to social factors. We select the web-log (blog) entries from blogs focused at different interest groups, assign these entries a politeness score on a scale of 0 to 1 using our classifiers, and compare these scores.

II. Background

The meaning of politeness and concomitant concepts, and the claims for universals have shown considerable divergence and lack of clarity as they have received increased attention since Brown and Lewinson's proposed framework [1], [3], [4]. Scholars use a variety of approaches to an account of politeness: the social-norm view, the conversational-maxim view; the face-saving view; and the conversational-contract view [5]. While none of these views is considered adequate, the face-saving view by Brown and Levinson is seen as the most clearly articulated and is the most popular.

Brown and Levinson contend that linguistic politeness must be communicated, that it constitutes a message. They assert that the failure to communicate the intention to be polite may be taken as absence of the required polite attitude. They propose a framework to explain politeness in which their rational Model Person has 'face', the individual's self-esteem. This face is a culturally elborated public self-image that every member of a society wants to claim for himself [5]. They characterize two types of face in terms of participant wants rather than social norms:

Negative Face: "the want of every 'competent adult member' that his action be unimpeded by others"

Positive Face: "the want of every member that his wants be desirable to at least some others"

The organizing principle for their politeness theory is the idea that "some acts are intrinsically threatening to face and thus require softening ..." To this end, each group of language users develops politeness principles from which they derive certain linguistic strategies. It is by the use of these so-called politeness strategies that speakers succeed in communicating both their primary message(s) as well as their intention to be polite in doing so. And in doing so, they reduce the face loss that results from the interaction.

The choice of a specific linguistic form is to be viewed as a specific realization of one of the politeness strategies in light of the speaker's assessment of the utterance context. Brown and Levinson outline four main types of politeness strategies: bald on-record, negative politeness, positive politeness, and off-record (indirect). The speaker must choose a linguistic means that will satisfy the strategic end. Since each strategy embraces a range of degrees of politeness, the speaker will be required to consider the specific linguistic forms used and their overall effect when used in conjunction with one another.

We try to identify such strategies and use them to construct our classifiers. A brief description of the classifiers we used is given below.

1) *Naive Bayes*: Naive Bayes is a highly practical learning method whose performance is shown to be comparable to that of neural network and decision tree learning in some domains. It applies to the learning tasks where each instance x is described by a conjunction of attribute values and where the target function $f(x)$ can take on any value from some finite set V. A set of training examples of the target function is provided, and a new instance is presented, described by the tuple of attribute values $<a_l, a_2, ..., a_n>$. The learner is asked to predict the target value, or classification, for this new instance. The Bayesian approach to classifying the new instance is to assign the most probable target value, V_{MAP}, given the attribute values $<a_l, a_2, ..., a_n>$ that describe the instance [6].

The naive Bayes classifier is based on the simplifying assumption that the attribute values are conditionally independent given the target value. In other words, the assumption is that given the target value of the instance, the probability of observing the conjunction $a_l, a_2, ..., a_n$ is just the product of the probabilities for the individual attributes.

2) *Naive Bayes Multinomial*: In the multinomial model, a document is an ordered sequence of word events, drawn from the same vocabulary V. We assume that the lengths of documents are independent of class. We again make a similar naive Bayes assumption: that the probability of each word event in a document is independent of the words context and position in the document. Thus, each document d_i is drawn from a multinomial distribution of words with as many independent trials as the length of d_i. This yields the familiar "bag of words" representation for documents [7]. Whereas simple naive Bayes would model a document as the presence and absence of particular words, multinomial naive bayes explicitly models the word counts and adjusts the underlying calculations to deal with in.

3) *J48*: J48 is an open source Java implementation of the C4.5 algorithm in the weka data mining tool. C4.5 is an algorithm used to generate a decision tree developed by Ross Quinlan. C4.5 is an extension of Quinlan's earlier ID3 algorithm. The decision trees generated by C4.5 can be used for classification, and for this reason, C4.5 is often referred to as a statistical classifier. At each node of the tree, C4.5 chooses the attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. The splitting criterion is the normalized information gain (difference in entropy). The attribute with the highest normalized information gain is chosen to make the decision. The C4.5 algorithm then recurses on the smaller sublists [8].

4) *Random Forest*: Random Forests grows many classification trees. To classify a new object from an input vector, put the input vector down each of the trees in the forest. Each tree gives a classification, and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest).

Each tree is grown as follows:

1) If the number of cases in the training set is N, sample N cases at random - but with replacement, from the original data. This sample will be the training set for growing the tree.
2) If there are M input variables, a number m<<M is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. The value of m is held constant during the forest growing.
3) Each tree is grown to the largest extent possible. There is no pruning.

In the original paper on random forests, it was shown that the forest error rate depends on two things:

1) The correlation between any two trees in the forest. Increasing the correlation increases the forest error rate.
2) The strength of each individual tree in the forest. A tree with a low error rate is a strong classifier. Increasing the strength of the individual trees decreases the forest error rate.

Reducing m reduces both the correlation and the strength. Increasing it increases both. Somewhere in between is an "optimal" range of m - usually quite wide. Using the oob error rate a value of m in the range can quickly be found. This is the only adjustable parameter to which random forests is somewhat sensitive [9].

5) *IBk*: This is an implementation of the k-nearest neighbors algorithm. This basic instance-based algorithm assumes all instances correspond to points in the n-dimensional space. The nearest neighbors of an instance are defined in terms of the standard Euclidean or Manhattan distance. The value of the classification label for an input x returned by this algorithm is just the most common value of label among the k training examples nearest to x [6].

6) *SMO*: Sequential minimal optimization (SMO) is an algorithm for solving the optimization problem which arises during the training of support vector machines. It was invented by John Platt in 1998 at Microsoft Research. SMO is widely used for training support vector machines and is implemented by the popular LIBSVM tool. SMO breaks the optimization problem in SVM into a series of smallest possible subproblems, which are then solved analytically [10].

III. Experiments

Our training data is from two different domains:

1) Wikipedia
2) Stack Exchange

We experiment on two different types of classifiers:

1) Bag of Words classifier (BOW)
2) Linguistically Informed classifer (Ling.)

For Linguistically Informed classifer (Ling.), we use the following features:

1) Gratitude
2) Deference
3) Greeting
4) Positive lexicon
5) Negative lexicon
6) Apologizing

7) Please
8) Please start
9) Indirect (btw)
10) Direct question
11) Direct start
12) Counterfactual modal (Could/Would)
13) Indicative modal (Can/Will)

We run experiments of two types:

1) In-domain: We use 5-fold cross-validations for these experiments. The experiments are:
   - Training on Wikipedia, Testing on Wikipedia
   - Training on Stack-Exchange, Testing on Stack-Exchange
2) Cross-domain:
   - Training on Wikipedia, Testing on Stack-Exchange
   - Training on Stack-Exchange, Testing on Wikipedia

The training and 5-fold cross-validation (In-domain) is done as follows:

1) Sort the training requests by their politeness scores.
2) Get top 25% of requests, and label them as positive.
3) Get bottom 25% of requests, and label them as negative.
4) Divide the data into 80% for training and 20% for testing.
5) Run classifier training procedure on training data.
6) Test the classifier on testing data.
7) Go back to Step 4 to repeat the procedure for different sets of training and testing data, and then take the average performance.

For cross-domain experiments, we train the classifiers again using Steps 1-3 above. We use the alternate domain data for testing.

For each experiment type and classifier type, we have four sets of experiments:

1) Using String-to-word unsupervised filter with alphabetic tokenizer (pre_alpha)
2) Using String-to-word unsupervised filter with alphabetic tokenizer followed by attribute selection (pre_alpha_with_attribute_selection)
3) Using String-to-word unsupervised filter with word tokenizer (pre_word)
4) Using String-to-word unsupervised filter with word tokenizer followed by attribute selection (pre_word_with_attribute_selection)

We use the following settings with String-to-word unsupervised filter:

- IDFTransform: True
- TFTransform: True
- attributeIndices: first-last
- doNotOperateOnFirstClassBasis: False
- invertSelection: False
- lowerCaseTokens: False
- minTermFrequency: 10
- normalizeDocLength: No Normalization
- outputWordCounts: True
- periodicPruning: -1.0
- stemmer: NullStemmer
- stopwords: weka-3-6-10
- useStoplist: False
- wordsToKeep: 1000

For attribute selection, we use:

- evaluator: InfoGainAttributeEval and
- search: Ranker with threshold 0.0

In each experiment set, we collect experiment results on these classifiers:

1) Naive Bayes
2) Naive Bayes Multinomial
3) J48
4) Random Forest with:
   - 10 trees
   - 100 trees
5) IBk (Instance-based k), the K-nearest neighbours classifier with:
   - K=1 and using Euclidean distance
   - K=10 and using Euclidean distance
   - K=1 and using Manhattan distance
   - K=10 and using Manhattan distance
6) SMO (Support vector classifier)

We observe that linguistically informed classifiers (Ling.) using String-to-word unsupervised filter with alphabetic tokenizer followed by attribute selection (pre_alpha_with_attribute_selection) generally give the best in-domain and cross-domain results. We use the classifiers to determine politeness in some blogs. We've used the blog entries from the following blogs:

1) blog 1: http://blogs.wsj.com/peggynoonan/, A Wall Street Journal Columnist
2) blog 2: http://www.thefashionpolice.net/, A blog about shopping and style
3) blog 3: http://www.rogerebert.com/reviews/tyler-perrys-a-madea-christmas-2013 , A blog for Movie reviews
4) blog 4: http://www.thedailybeast.com/articles/2013/12/17/it-s-conservatives-who-really-want-christ-out-of-christmas.html, A blog dedicated to breaking news and sharp commentary
5) blog 5: http://www.blogcatalog.com/blogs/haters-be-hatin, A satirical humour blog

IV. Experimental Results

This section describes the results for the experiments.

A. In-domain Experiments

Four sets of experiments are done for in-domain analysis using a 5-fold cross-validation.

The correctly classified instances (by %) for In-domain analysis on Wikipedia requests using Bag of Words classifiers are shown in table I.

The correctly classified instances (by %) for In-domain analysis on Wikipedia requests using Linguistic classifiers are shown in table II.

The correctly classified instances (by %) for In-domain analysis on Stack Exchange requests using Bag of Words classifiers are shown in table III.

The correctly classified instances (by %) for In-domain analysis on Stack Exchange requests using Linguistic classifiers are shown in table IV.

B. Cross-domain Experiments

Four sets of experiments are done for cross-domain analysis using a 5-fold cross-validation.

The correctly classified instances (by %) for Cross-domain analysis with Wikipedia requests for training and Stack Exchange requests for testing and using Bag of Words classifiers are shown in table V.

The correctly classified instances (by %) for Cross-domain analysis with Wikipedia requests for training and Stack Exchange requests for testing and using Linguistic classifiers are shown in table VI.

The correctly classified instances (by %) for Cross-domain analysis with Stack Exchange requests for training and Wikipedia requests for testing and using Bag of Words classifiers are shown in table VII.

The correctly classified instances (by %) for Cross-domain analysis with Stack Exchange requests for training and Wikipedia requests for testing and using Linguistic classifiers are shown in table IX.

C. Experiments on web logs

The classification results using Wikipedia requests for training are shown in table **??**

V. Related Work

VI. Conclusions and Future Work

References

[1] A. J. Meier. *Defining Politeness: Universality in Appropriateness*

[2] Cristian Danescu-Niculescu-Mizil, Moritz Sudhof, Dan Jurafsky, Jure Leskovec and Christopher Potts. *A computational approach to politeness with application to social factors*.

[3] BROWN, P. and LEVINSON, S. 1978. *Universals in Language Usage: Politeness Phenomena*. In E. Goody (ed.), Questions and Politeness, 56-289. Cambridge University Press, Cambridge.

[4] BROWN, P. and LEVINSON, S. 1987 *Politeness: Some Universals in Language Usage*. Cambridge University Press, Cambridge.

[5] Bruce Fraser. *Perspectives on Politeness*

[6] Tom Mitchell. *Machine Learning* McGraw-Hill Science/Engineering/Math; 1 edition (March 1, 1997)

[7] Andrew McCallum and Kamal Nigam. *A Comparison of Event Models for Naive Bayes Text Classification*

[8] http://en.wikipedia.org/wiki/C4.5_algorithm

[9] www.stat.berkeley.edu/ breiman/RandomForests/cc_home.htm

[10] http://en.wikipedia.org/wiki/Sequential_minimal_optimization

TABLE I: In-domain analysis on Wikipedia requests using Bag of Words classifiers

Classifier	pre_alpha	pre_word	pre_alpha_with_attribute_selection	pre_word_with_attribute_selection
Naive Bayes	74.5864%	74.4945%	77.2518%	77.068%
Naive Bayes Multinomial	78.9063%	79.6415%	80.5147%	80.1471%
J48	70.864%	71.2776%	73.7132%	74.3107%
Random Forest (10 trees)	74.5404%	73.6673%	76.7004%	76.6085%
Random Forest (100 trees)	80.7445%	80.193%	80.3309%	79.8254%
iBK (k=1, using Euclidean Distance)	64.8897%	64.1544%	71.2316%	70.6342%
iBK (k=10, using Euclidean Distance)	59.1912%	58.9154%	76.7463%	76.7923%
iBK (k=1, using Manhattan Distance)	63.2813%	63.1434%	71.2316%	69.1636%
iBK (k=1, using Manhattan Distance)	56.4338%	56.1581%	74.6783%	73.4835%
SMO	80.193%	79.8713%	82.307%	82.2151%

TABLE II: In-domain analysis on Wikipedia requests using Linguistic classifiers

Classifier	pre_alpha	pre_word	pre_alpha_with_attribute_selection	pre_word_with_attribute_selection
Naive Bayes	74.4485%	74.7702%	77.0221%	76.3327%
Naive Bayes Multinomial	80.7904%	80.239%	80.4688%	80.3309%
J48	72.7022%	72.4265%	75%	73.3456%
Random Forest (10 trees)	72.932%	74.7702%	76.7004%	77.4357%
Random Forest (100 trees)	79.9173%	80.6066%	80.1011%	80.4228%
iBK (k=1, using Euclidean Distance)	64.6599%	64.8438%	71.4614%	71.829%
iBK (k=10, using Euclidean Distance)	60.2022%	59.6967%	76.5165%	76.7923%
iBK (k=1, using Manhattan Distance)	64.6599%	64.8897%	70.5423%	70.5423%
iBK (k=1, using Manhattan Distance)	59.4669%	59.5129%	74.6783%	74.9081%
SMO	81.3879%	80.3768%	82.2151%	81.0202%

TABLE III: In-domain analysis on Stack Exchange requests using Bag of Words classifiers

Classifier	pre_alpha	pre_word	pre_alpha_with_attribute_selection	pre_word_with_attribute_selection
Naive Bayes	68.4%	67.7%	71.8%	71.2%
Naive Bayes Multinomial	71.8%	71.55%	72.35%	71.4%
J48	67.15%	65.9%	69.05%	69.75%
Random Forest (10 trees)	69.5%	68.75%	70.15%	69.95%
Random Forest (100 trees)	73.6%	73.45%	72.35%	72.45%
iBK (k=1, using Euclidean Distance)	57.7%	58.85%	65.75%	65.6%
iBK (k=10, using Euclidean Distance)	53.2%	53.15%	66.95%	66.35%
iBK (k=1, using Manhattan Distance)	56.9%	56.85%	65.9%	63.55%
iBK (k=1, using Manhattan Distance)	51.35%	51.95%	64.6%	63.1%
SMO	74.55%	73.5%	74.8%	75.05%

TABLE IV: In-domain analysis on Stack Exchange requests using Linguistic classifiers

Classifier	pre_alpha	pre_word	pre_alpha_with_attribute_selection	pre_word_with_attribute_selection
Naive Bayes	68.45%	68.55%	72.4%	72.4%
Naive Bayes Multinomial	72.85%	72.7%	74.6%	74.4%
J48	67.7%	67.25%	71.1%	69.65%
Random Forest (10 trees)	69.15%	67.35%	71.6%	70.25%
Random Forest (100 trees)	74.2%	74.15%	73.35%	72.75%
iBK (k=1, using Euclidean Distance)	58.4%	59.2%	64.9%	63.5%
iBK (k=10, using Euclidean Distance)	55.25%	57.65%	71.2%	71.15%
iBK (k=1, using Manhattan Distance)	58.3%	58.6%	63.8%	63.15%
iBK (k=1, using Manhattan Distance)	53.55%	54.45%	68.8%	67.85%
SMO	72.95%	73.95%	75%	75.95%

TABLE V: Cross-domain analysis with Wikipedia requests for training and Stack Exchange requests for testing and using Bag of Words classifiers

Classifier	pre_alpha	pre_word	pre_alpha_with_attribute_selection	pre_word_with_attribute_selection
Naive Bayes	63.1%	62.7%	65.35%	64.85%
Naive Bayes Multinomial	66.45%	66%	66.55%	66.5%
J48	62.55%	61.6%	60.4%	61.1%
Random Forest (10 trees)	64.85%	64.95%	64.05%	64.35%
Random Forest (100 trees)	66.2%	66.25%	64.65%	64.65%
iBK (k=1, using Euclidean Distance)	55.05%	55%	62.6%	63.25%
iBK (k=10, using Euclidean Distance)	50.45%	50.25%	61.15%	61.35%
iBK (k=1, using Manhattan Distance)	54.7%	54.3%	61.05%	60.55%
iBK (k=1, using Manhattan Distance)	50.5%	50.35%	58.35%	58.75%
SMO	64.65%	65.55%	65.35%	64.4%

TABLE VI: Cross-domain analysis with Wikipedia requests for training and Stack Exchange requests for testing and using Linguistic classifiers

Classifier	pre_alpha	pre_word	pre_alpha_with_attribute_selection	pre_word_with_attribute_selection
Naive Bayes	64.4%	64.35%	65.55%	65.55%
Naive Bayes Multinomial	66.3%	66%	66.3%	66.5%
J48	61.45%	61.2%	61.05%	60.85%
Random Forest (10 trees)	63.95%	62.3%	65.1%	63.45%
Random Forest (100 trees)	62.85%	63.65%	64.65%	64.65%
iBK (k=1, using Euclidean Distance)	56.75%	56.75%	63.35%	62.5%
iBK (k=10, using Euclidean Distance)	51.85%	51.9%	60.4%	60.7%
iBK (k=1, using Manhattan Distance)	55.15%	55.65%	60.1%	59.6%
iBK (k=1, using Manhattan Distance)	51.35%	50.95%	58.05%	59.35%
SMO	64.9%	64.95%	65.8%	65.45%

TABLE VII: Cross-domain analysis with Stack Exchange requests for training and Wikipedia requests for testing and using Bag of Words classifiers

Classifier	pre_alpha	pre_word	pre_alpha_with_attribute_selection	pre_word_with_attribute_selection
Naive Bayes	60.9375%	61.1213%	66.0386%	65.3493%
Naive Bayes Multinomial	68.9338%	68.75%	65.4412%	65.579%
J48	62.1783%	62.546%	64.0625%	64.7518%
Random Forest (10 trees)	66.9577%	64.568%	66.682%	67.6471%
Random Forest (100 trees)	70.5423%	68.9338%	68.1985%	68.4743%
iBK (k=1, using Euclidean Distance)	57.1691%	57.5827%	62.5919%	62.9596%
iBK (k=10, using Euclidean Distance)	53.3088%	52.8952%	66.5441%	66.4522%
iBK (k=1, using Manhattan Distance)	56.5257%	56.296%	61.2592%	61.6728%
iBK (k=1, using Manhattan Distance)	52.0221%	52.0221%	64.9816%	64.8897%
SMO	71.0938%	71.3235%	68.704%	68.75%

TABLE VIII: Cross-domain analysis with Stack Exchange requests for training and Wikipedia requests for testing and using Linguistic classifiers

Classifier	pre_alpha	pre_word	pre_alpha_with_attribute_selection	pre_word_with_attribute_selection
Naive Bayes	60.8915%	60.9835%	65.3952%	64.568%
Naive Bayes Multinomial	69.761%	69.6691%	68.0147%	68.2904%
J48	62.9136%	60.6618%	65.7169%	65.2114%
Random Forest (10 trees)	64.0625%	61.9945%	65.3952%	64.0625%
Random Forest (100 trees)	70.0368%	69.0257%	67.4632%	66.9577%
iBK (k=1, using Euclidean Distance)	58.1801%	57.8125%	57.5827%	58.1342%
iBK (k=10, using Euclidean Distance)	55.239%	53.3548%	63.1434%	62.8676%
iBK (k=1, using Manhattan Distance)	58.7776%	57.307%	57.5368%	57.9963%
iBK (k=1, using Manhattan Distance)	53.7224%	53.171%	64.1544%	64.0165%
SMO	70.9099%	71.6452%	69.761%	69.4393%

TABLE IX: Classification results using Wikipedia requests for training

Classifier	Blog 1		Blog 2		Blog 3		Blog 4		Blog 5	
	polite	impolite	polite	impolite	polite	impolite	polite	impolite	polite	impolite
Naive Bayes	60.8915%	0	60.9835%	0	65.3952%	0	64.568%	0	123	0