Comparison of Sentence-level Paraphrasing Approaches for Statistical Style Transformation

Foaad Khosmood
Department of Computer Science, California Polytechnic State University, San Luis Obispo, California, USA

***Abstract*-- We present six distinct English-language sentence paraphrasing techniques as transforms: *Active-to-Passive, Diction, Nodebox, Phrase, Simplify* and *Pivot translation*. We describe the basic algorithm for each and score them on precision and recall using 50 random sentences from project Tatoeba [1]. Our aim is to use these techniques for style transformation.**

Keywords: Computational Stylistics, Style Processcing, Natural Langauge Processing, Machine Learning, Computational Linguistics

1 INTRODUCTION

Statistical style transformation refers to an automatic method of rewriting text such that the information content remains the same, but the style of writing changes. Manual style transformation has been done successfully in the past [2]. In the absence of a precise universally accepted nomenclature and litmus test for different writing styles in natural languages, we define style as choice of expression [3] such as that often associated with an author. As the authorship attribution literature has shown, there are distinguishable style markers that can be used to successfully classify works by different authors [4][5] and hence also isolate their "style". Our approach uses this type of classification as a litmus test for achieving style transformation. This was done by Juola [6] already to evaluate manual style obfuscation cases.

Our model requires sentence level paraphrases to be continuously generated by modules (called transforms) and scored based on a greedy algorithm. Whichever paraphrased sentence would move the entire text statistically closest to the target style, that sentence is adopted and becomes part of the new text, but it could still be subject to further substitutions.

2 TRANSFORMS

The transforms are implemented as Python classes. They use various techniques such as typed dependency reformulation [7] or web-based validation [8] to produce a grammatical and synonymous sentence to the one they were passed in. All the implementation and most of the algorithms come from the authors, but external resources have been used quite extensively as stated below. We briefly describe each transforms.

2.1 Active to Passive transform (A2P)

Example: *The boy kicked the ball. => The ball was kicked by the boy.*

This transform converts an active voice sentence into its passive form. We encode 20 highly general rules in terms of parts-of-speech for basic active voice and equivalent passive voice cases [9][10]. The initial rules are mutated to produce 120 rules, adding variations to handle particles, adverbs and complex noun phrases. The input sentence is parsed into chunks using the Link Grammar Parser and compared with all the patterns in the database, and if matched, converted accordingly. Various NLP techniques are used to ensure plurals, pronouns and subject/verb agreement.

2.2 Diction Transform

Example: *The fact is, in the majority of cases, X is defined as an unknown. => Usually X is an unknown.*

This transforms employs a large number of rules for substituting wordiness and "bad style" phrases with their equivalent. The data sources are:

1. 655 rules from GNU *Diction* v. 1.11, itself based on *Elements of Style* [11].
2. 658 "wordiness" rules from Steve Hanov (Stevehanov.ca) partially based on William Zinsser's *On Writing Well.*
3. 409 word/alternative pairs from *The A to Z of alternative words* by the Plain English Campaign.

The rules are applied as straight string substitutions. Many of the rules had to be modified manually by the authors to remove "hints" and discussion originally designed to be examined by users of word editing programs. Some generic suggestions with no clear substitution suggestions were removed altogether from the original set of rules.

2.3 Nodebox transform

Example: *We recieved 6533 boxes. => We received thousands of boxes.*

This transform is based on the Nodebox Linguistics package [12]. It performs two tasks in particular: First, it

changes a set of unambiguous misspelled words, to their correct equivalents. Second, it converts Arabic numerical symbols modifying certain noun phrases into English approximation phrases.

2.4 Phrase replacement transform (Phrase)

Example: *I wanted to be with you alone => I desired to be with you only.*

This transform replaces phrases (defined as one or more consecutive words) with synonyms. It uses the Stanford Part of Speech Tagger to label each token in the phrase, and then uses Word Net to find synonyms for each non-overlapping phrase matching the POS in the input sentence.

For each phrase, this transform determines all the synsets of the same sense using *jcn* distance [13] to do Word Sense Disambiguation (WSD). The remaining candidates are ranked using two criteria: probability of co-occurrence with other words in the same sentence in the Microsoft BING N-Gram database (all bigrams and trigrams crawled by BING search engine) [14], and the probability of co-occurrence with other words in the same sentence in the target corpus. If the latter is not available, then only the search engine data is used.

Once a suitable candidate is chosen, NLP tools are used to produce the correct inflection before actual substitution. Heavy use of NLTK NLP tools [15] was made for this transform. The following figure summarizes the major steps in this transform and external tools used to accomplish them.

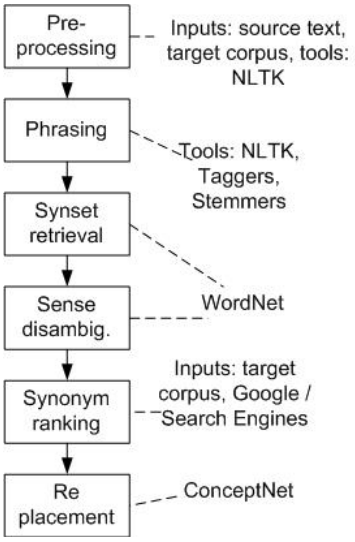

Figure 1. Phrase Transform

2.5 Simplify transform

Example: *I love Luna who is very cute and who is my pet. => I love Luna. Luna is very cute. Luna is my Pet.*

This transform aims to simplify sentences on several criteria, possibly producing more than one sentence as a result. It is written by Advaith Siddharthan [7] mostly in Perl, using the Stanford parser and the RASP toolkit. Two additional rules and a Python wrapper are written by the authors to integrate it into our evaluation system.

Simplify produces typed dependency trees for each sentence and then runs its user-modifiable rules to match patterns and adjust the final word order which could result in more than one sentence. It examines conjunctions and anaphora and looks for independent clauses that can be separated into their own sentences.

2.6 Pivot translation transform

Example: *I like to play Tennis. => I enjoy playing Tennis.*

This transform performs statistical translation by pivot: It translates an English sentence into another language and then translates the result back to English. The idea is that automatic translation tools usually produce valid but slightly modified sentences. A translation from English into another language and back to English could provide a valid semantic variation of the original with sufficient quality. Because of the high likelihood that the very same string is retuned, this transform accepts an ordered list of languages to try one at a time until a response different than the input is found.

As the back-end statistical translator, it supports Yahoo/Babelfish, Google Translator, and Microsoft Translator. We found the Microsoft product to be the most suitable for several reasons and we used it for this project.

3 METHODOLOGY

Fifty random English sentences are selected from the project Tatoeba [1] corpus, and passed on to each transform. Then each of the transforms produces responses consisting of zero, one or more paraphrased sentences for each of the input sentences. Some transforms such as *Phrase*, can produce more than one alternative paraphrase. Others such as *Diction* are highly specialized applying to few random sentences. We produce 381 paraphrases for the original 50 sentences. Each of these is scored on the scale of 0 to 3 by three separate human judges based on the following criteria.

Table 1. Evaluation criteria

Code	Criteria
3	Sentence is modified, preserves original meaning and original level of grammaticality.
2	Sentence is unmodified, or barely modified with very minor changes, or modified and the result is “acceptable” but not great.
1	Sentence is modified and the result is not acceptable.
0	No response, machine error or incomprehensible response.

4 RESULTS AND CONCLUDING REMARKS

Before the *Pivot translation* transform can be evaluated, we must decide which of the supported languages to use and how many languages to translate to before returning to English. Through a separate process using human 2 judges, we evaluate over 900 sentences using translation by pivot, and determined an ordered set of 8 languages (out of a total of 32 supported by Microsoft Translator) to use for the translation transform. In order of decreased effectiveness, they are: Czech, Swedish, Danish, Vietnamese, Norwegian, Dutch, Portuguese, and Spanish.

In addition, we generate three baseline translation "tours" (multiple language translations before returning to English) for comparison purposes. Tour1 is English->French->Spanish->German->English. Trou2 is English->Danish->Portuguese->Swedish->Vietnamese->English. TourR consists of English and four random languages.

All 381 paraphrases are scored for precision (majority score of the 3 judges). We derive the relevant precision and recall statistics as follows.

- *"recall" is calculated by counting the number of sentences for which the transform had a response (other than the input), divided by the total number of sentences, 50.*
- *"precision" is the average of all the (0-3) scores given by human evaluators to any response of the transform, across all sentences.*
- *"F-measure" is defined as* $F = \frac{2*(precision*recall)}{precision+recall}$

Table 2. Transform experiment evaluation results in terms of precision, recall and F-measure

Transform	Recall (%)	Precision (%)	F-measure
A2P	10	80	0.178
Diction	4	100	0.077
Nodebox	20	87	0.325
Phrase	100	82	0.903
Simplify	7.6	78	0.138
Translation	74	83	0.782
Tour 1	88	63	0.733
Tour 2	78	70	0.738
Tour R	88	58	0.702

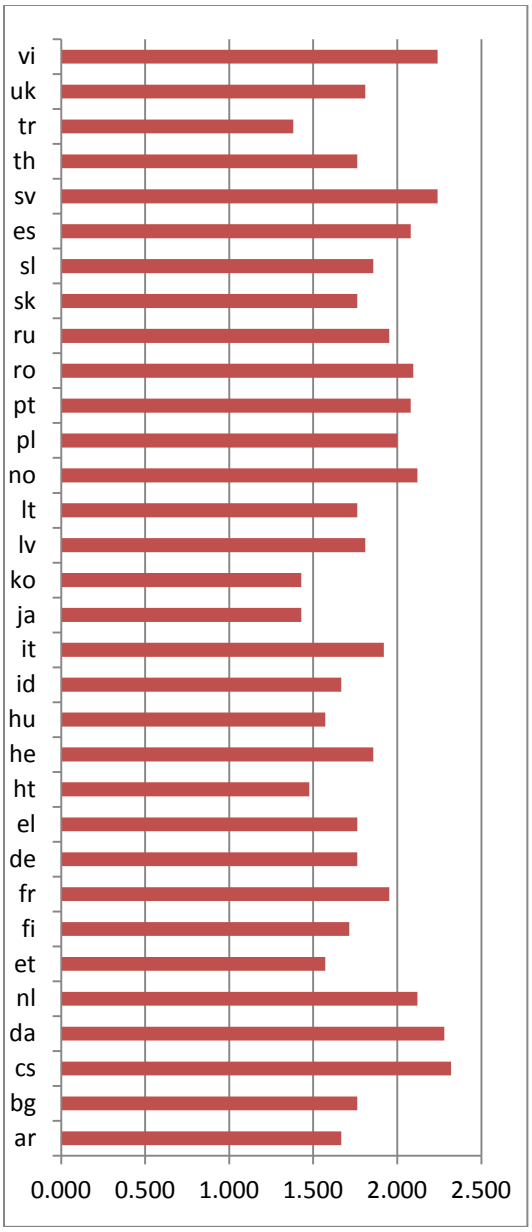

Figure 2. Microsoft translator language performance

As observed from Fig. 3, the transforms can be generally divided into two groups: high precision and high recall. The transforms in the high precision group have unsurprisingly low recall. This is because the patterns they generally seek are very specific, such as the "wordiness" phrases in *Diction*. The high recall group has more general pattern matching. Every sentence contains at least one word that would have a synonym in Word Net. Almost every sentence would have at least one translated paraphrase from some language.

We find that Phrase has the highest F-measure. But the real goal for our work is not to find a single approach, but to combine the power of different approaches to get a super-

transform of increased recall and precision. We are pursuing future work in this area.

Table 3. Microsoft Translator supported language codes (ISO 639-2)

Code	Language	Code	Language	Code	Language
ar	Arabic	he	Hebrew	ro	Romanian
bg	Bulgarian	hu	Hungarian	ru	Russian
cs	Czech	id	Indonesian	sk	Slovak
da	Danish	it	Italian	sl	Slovenian
nl	Dutch	ja	Japanese	es	Spanish
et	Estonian	ko	Korean	sv	Swedish
fi	Finnish	lv	Latvian	th	Thai
fr	French	lt	Lithuanian	tr	Turkish
de	German	no	Norwegian	uk	Ukrainian
el	Greek	pl	Polish	vi	Vietnamese
ht	Haitian	pt	Portuguese		

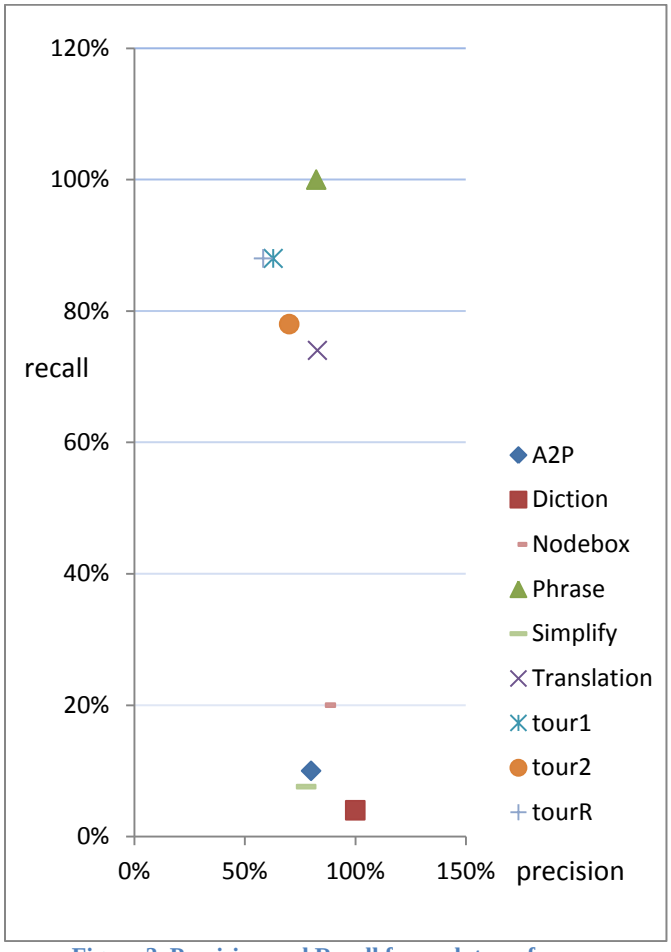

Figure 3. Precision and Recall for each transform

5 REFERENCES

[1] Project Tatoeba, English language sentence corpus (about 150,000 sentences). accessed from tatoeba.org, September 2010.

[2] D. L. Hoover. *Language and style in* The Inheritors, University Press of America. 1999.

[3] Jane Walpole. "Style as Option," *College Composition and Communication*, vol. 31, No. 2, Recent Work in Rhetoric: Discourse Theory, Invention, Arrangement, Style, Audience. May, 1980. pp. 205-212.

[4] J. F. Burrows. 'Delta': A Measure of Stylistic Difference and a Guide to Likely Authorship'. *Literary and Linguistic Computing*. 17(3), 2002. pp. 267-87.

[5] D. L. Hoover. 'Testing Burrows's Delta'. *Literary and Linguistic Computing*. 19(4), 2004. pp. 453-75.

[6] Patrick Juola. "Empirical evaluation of authorship obfuscation using JGAAP," in *Proceedings of the 3rd ACM workshop on Artificial intelligence and security*, AISec, 2010.

[7] Advaith Siddharthan. "Complex lexico-syntactic reformulation of sentences using typed dependency representations". In Proceedings of the 6th International Natural Language Generation Conference (INLG 2010), Dublin, Ireland. pp. 125-133.

[8] Houda Bouamor et. al. "Web-based validation for contextual targeted paraphrasing," *Proceedings of the ACL Workshop on Monolingual Text-To-Text Generation* 2011.

[9] EnglishPage by Language Dynamics, (www.englishpage.com), accessed Dec. 2010.

[10] Huddleston and Pullum. *A Student's Introduction to English Grammar*, Cambridge University Press, 2005.

[11] William Strunk. *The elements of style*, Ithaca, N.Y.: Priv. print., 1918.

[12] Frederik De Bleser, Tom De Smedt, Lucas Nijs. NodeBox version 1.9.5 for Mac OS X. Retrieved March 2010, from: http://nodebox.net

[13] Jiang, Jay and Conrath, David (1997). Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of International Conference Research on Computational Linguistics* (ROCLING X), Taiwan.

[14] Micorsoft Web N-Gram service, public Beta program, http://web-ngram.research.microsoft.com/info/ accessed 2011.

[15] Steven Bird, Ewan Klein, and Edward Loper, *Natural Language Processing with Python*, O'Reilly Media. 2009.