2013 IEEE Seventh International Conference on Semantic Computing

Structural Parse Tree Features for Text Representation

Sean Massung ChengXiang Zhai Julia Hockenmaier

Department of Computer Science, College of Engineering
University of Illinois at Urbana-Champaign
{massung1,czhai,juliahmr}@illinois.edu

***Abstract*—We propose and study novel text representation features created from parse tree structures. Unlike the traditional parse tree features which include all the attached syntactic categories to capture linguistic properties of text, the new features are solely or primarily defined based on the tree structure, and thus better reflect the pure structural properties of parse trees. We hypothesize that these new complex structural features capture an orthogonal perspective of text even compared to advanced syntactic ones. Evaluation based on three different text categorization tasks (*i.e.*, nationality detection, essay scoring, and sentiment analysis) shows that the proposed new tree structure features complement the existing ones to enrich text representation. Experiment results further show that a combination of the proposed new structure features with word n-grams can improve F_1 score and classification accuracy.**

I. Introduction

Text representation is a fundamental issue in many text processing applications such as information retrieval, text clustering, and text categorization. Text representation is also critical for generating useful features to be used in many machine learning algorithms to support natural language processing applications. In general, we can represent text by a set of extracted features; the simplest features can be just the words in the text.

The issue of text representation is complicated because different tasks tend to require a somewhat different perspective of representation—thus a different feature set. For example, while functional words are generally not useful for topic categorization, they may be useful for the author attribution categorization task, which may also benefit from features capturing sentence structures. It is therefore important to develop a rich set of potential features that can represent text from different perspectives and to understand what kind of features are most effective for which tasks.

By far, the most common way to generate features is to segment text into words and record their n-grams; indeed, unigrams are quite common for information retrieval and text classification applications.

In addition to content features, functional words and syntactic features have also been considered, notably for tasks such as author attribution or essay scoring. Complementary with content features, syntactic features can better reflect the writing style of an article. For example, simple syntactic features such as n-grams of part-of-speech tags and unigram function words were used for authorship attribution in [1] and [2]. To further capture syntactic structures, grammatical productions (rewrite rules) were also discussed as potential features in [3], where the authors showed that rule frequencies were significantly different across classes and used them as features in some simple classification tasks. Later work used syntactic tree features for scoring non-native speech [4], authorship attribution in [5] and [6], deception detection in [7], relation extraction in [8], and even tree kernel methods in [9] and [10].

In this paper, we propose to investigate a new dimension of text representation based on parse trees with more emphasis on structural representation. Specifically, we propose to define structural features solely based on structural properties of a parse tree by ignoring all of the syntactic categories in the tree. More specifically, we call such new features skeletons to indicate their emphasis on pure structures rather than rewrite rules. A skeleton is defined as any subtree of a parse tree without including any syntactic categories. Compared with syntactic rewrite rules, skeletons can better capture the structural properties of a whole parse tree. Indeed, an important advantage of skeletons over regular syntactic features is that they can capture "global tree structures" without causing problems of data sparseness or overfitting. Because of the focus on pure structures, even relatively large skeletons can be observed multiple times in a reasonably large set of text articles; in contrast, if we are to attach the syntactic categories, we would end up having far more specialized features that may not be observed multiple times in a data set. We thus hypothesize that skeletons can capture a new additional dimension of text that cannot be easily captured by either content features or regular syntactic features, and thus may serve well as complementary features with the existing ones.

We evaluate the proposed skeleton-based features using three different categorization tasks that likely would benefit from structural representation of text: nationality detection, essay grading, and sentiment analysis. We compare feature combinations of the proposed new features with three common simple features (n-grams of words, part-of-speech tags, and function words). We also investigate existing tree features (rewrite rules, syntactic categories, production branching factors, and tree depth), showing that the new skeleton-based fea-

978-0-7695-5119-7/19 $26.00 © 5119 IEEE
DOI 10.1109/ICSC.2013.13

tures provide orthogonal information compared to the simpler features and validating their usefulness for text representation.

II. Related Work

Tree structure has been explored before, though not in a text representation perspective. A treebank described in [11] allows grammatical parse trees to be browsed based on structure alone, but does not provide any sort of classification component. In [12], the authors use dependency tree structure in a sentence similarity metric for textual entailment. A sentence similarity measure could possibly be generalized to an entire document, though a purely-structural sentence similarity measure has not been presented before.

Although both these works consider tree skeletons, they are not used as a feature for text representation, and thus have not been used as features for classification, clustering, or information retrieval.

In previous studies of features for text representation, the authors only examined a small subset of feature competitors: in [5], unigrams, bigrams, and trigrams of words; in [6], unigram and bigram part-of-speech (POS) tags and bag-of-words function words (FW); in [7], unigram and bigram words and unigram, bigram, and trigram parts-of-speech. In addition to a limited comparison set, each paper only considered one domain; authorship attribution in the first two, and deception detection in the second. In this paper, we compare these existing features with the proposed new features on three additional tasks: nationality detection, essay scoring, and sentiment analysis.

[4] examines tree features at a much higher level in the form of nonterminals per sentence, *e.g.* number of noun phrases per sentence and mean number of prepositional phrases per sentence. The work was mainly focused on investigating potential features so no classification tasks were performed.

[6] mines discriminative frequent PCFG tree patterns for each author. Features used were the rewrite rules and a new pattern, k-embedded-edge (ee) subtrees: subtrees that share a set of k ancestor-descendant subtrees. Therefore, a 0-ee subtree would be one arbitrarily-sized subtree, and a 1-ee subtree would be one subtree and one descendant subtree anywhere in the parse tree. This creates an exponential number of potential patterns, and the authors define algorithms in order to process this large amount of data before pruning the number of ee trees to be used as features. In fact, the algorithms were run on a petascale supercomputer, which justified the implication that their method is quite computationally intensive. Besides the concern of computational complexity, another concern is the high susceptibility of the large number of patterns to overfitting. In contrast, the skeleton features proposed in this paper are efficient to compute and systematically capture the major structures in a parse tree.

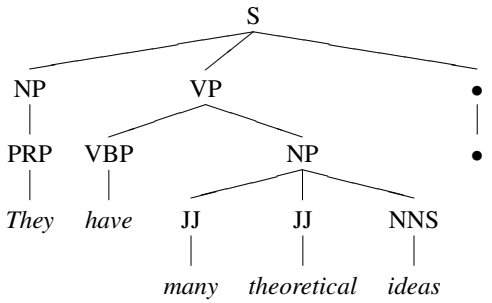
[8] explores feature extraction from sequence (words), syntactic (grammatical parse trees), and dependency (dependency parse trees) subspaces. Features used were n-grams for the word sequences, grammar productions for PCFGs, and dependency paths for the dependency parse trees. They concluded that adding all these features together versus separately only slightly increases performance. We suspect that this is because the structural information encoded in the parse trees is not taken into account.

Grammatical parse tree features have also been explored in classification tasks as tree kernels in [13], [15], and [14]. Again, none of this previous work takes into account the structure of the trees themselves, but rather focuses on the syntactic categories as the main avenue of information. [13] mainly focuses on reducing the feature space of "all subtrees" for the perceptron algorithm using rewrite-rule features. [15] explores a boosting method over "subtree stumps" (common subtree sequences). Finally, [14] provides yet another tree kernel method, focusing on efficient algorithms. They use parse tree substructures, a somewhat larger feature space than rewrite rules, but still consider only the node labels in addition to their order.

We evaluate our proposed skeleton features with three applications: nationality detection, essay scoring, and sentiment analysis. These tasks were previously studied in many papers, including, *e.g.*, [16], [17], [18], [19], and [20] . Our experiments show that the proposed new features can help improve performance for all these tasks, and potentially many others.

III. Existing Text Representation Methods

We now discuss existing text representation methods and their motivation. For this section and the next, a parse tree of the sentence "*They have many theoretical ideas.*" is used for examples and given below.



The parse tree is rooted with S, denoting $Sentence$; the sentence is composed of a noun phrase (NP) followed by a verb phrase (VP) and period. The leaves of the tree are the words in the sentence, and the preterminals (the direct parents of the leaves) are part-of-speech tags.

A. Words

Words are the simplest, most intuitive textual features. Although not a grammatical feature such as rewrite rules or function words, we include them because of their prevalence and accuracy. Besides, we expect words will be a very useful feature to combine with more advanced methods.

Stemming and stop word removal are optimizations often applied to word features. Both these methods greatly reduce

the feature space, omitting irrelevant words (stop word removal) and grouping words with the same meaning together, such as "player" and "playing" (stemming).

n-grams of words and all the following simple features are also collected and utilized. This is quite a common practice, with n-grams of words first described in [21].

B. Part-of-Speech Tags

Part-of-speech tags are a common grammatical feature. Their small, finite number lends them to be simple features for a classifier. When expanding to n-grams of part-of-speech tags, their small number also ensures that there are still a relatively low number of features generated (opposed, mainly, to words).

POS tags are perhaps the syntactic analog of basic words, in that they are simple and robust. They capture grammar usage at its most basic level. High accuracy POS taggers ($\geq 97\%$) ensure cleanly processed data.

C. Syntactic Categories

Syntactic category features can be thought of as an extension of POS tags to parse trees. This creates a distribution of non-terminal productions over each class. The trees are simply traversed, tallying the labels of internal nodes: S:1, NP:2, VP:1, PRP:1, VBP:1, JJ:2, and NNS:1.

The goal of syntactic categories is to observe phrase structure occurrence at a level above POS tags and words. This feature is similar to POS tags in that sense, and it even records them when examining nodes near the leaves of the tree. We hope this feature does at least as well as unigram POS tags, since they are a strict subset of this feature.

D. Function Words

Function words are a well-performing feature for authorship attribution as noted by [1] and [2]. They attempt to capture nuances in text that remain largely an unconscious byproduct of individual authors. In our experiments, we see if our 320 function words also distinguish between nationality, essay grades, or positive or negative sentiment.

Since the n-gram feature generation tools in our toolkit already existed for POS tags and words, we ran the function words collected from the text through this part of the system as well, mainly out of curiosity if bigram function words or higher turned out to be useful.

E. Tree Depth

Tree depth simply measures the maximum number of productions to travel to reach a leaf. In the example above, this would be four (including the word terminals). The depth is tallied for all trees in a label, creating a distribution of depths.

Since tree depth is approximately proportional to sentence length, it may prove to be a useful feature since sentence length is often used as a feature for distinguishing texts. Depth was explored in [4] and found to be positively correlated to scores in *spoken* exams.

F. Production Branch Factors

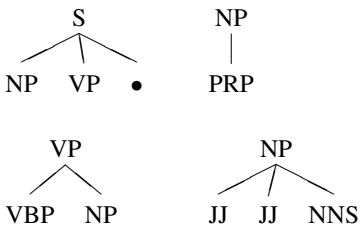
This creates a histogram of "branching factors" at each non-terminal. For example, the S production in the example has three productions, therefore contributing one to the branch count value "3". Remember, we ignore the terminal productions. In this feature case, it would inflate the counts for a branch factor of 1 anyway.

Statistics for the example parse are as follows: one count of one branch for NP→PRP; one count of two branches for VP→VBP, NP; and two counts for three branches in S→ ... and NP→ ...

Counting branches attempts to capture the complexity of a production (and the overall sentence). Generating more labels usually results in a longer, more involved sentence, perhaps indicative of specific classes.

G. Rewrite Rules

This method tallies subtrees from each sentence's parse and was one of the tree features in [6] and others. The following subtrees would be recorded from the example sentence *"They have many theoretical ideas."*:



This process is repeated for all sentences in all texts belonging to a given class, so each class has a distribution of these subtrees. It can be thought of as a "bag-of-trees" method.

This feature is desirable, as particular parse trees could be common for any particular category. For example, in age detection, more complicated tree structures could be scarce for younger writers. Similarly, authors whose native language is not English may only select sentence structures from a relatively small learned collection, or repeat similar practiced patterns.

IV. Novel Skeleton-Based Features

A. Skeletons

The skeleton method is a novel procedure that recursively descends into subtrees, recording the internal structure with disregard to internal node labels. This attempts to capture the flow or phrasal structure of sentences while being agnostic to actual labels.

The simple CountSkeletons function is described below. Skeleton returns the skeletal structure of the tree rooted at the parameter.

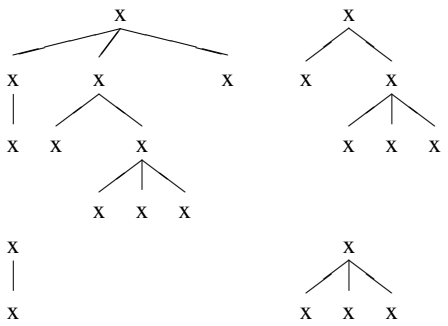
As with the previous feature, simple frequency counts are kept for each tree skeleton in each sentence in the entire class data set as indicated by the function IncrementCount. The

Algorithm 1 Counting different skeletons in a parse tree

procedure COUNTSKELETONS(T)
  $token \leftarrow$ SKELETON(T)
  INCREMENTCOUNT($token$)
  for each subtree $t \in T$ **do**
    COUNTSKELETONS(t)
  end for
end procedure

skeleton structure representations can be recorded as sets of parenthesis: `((())(()(()))())`.

For example, here are the skeletons generated from the sentence *They have many theoretical ideas*:



B. Annotated Skeletons

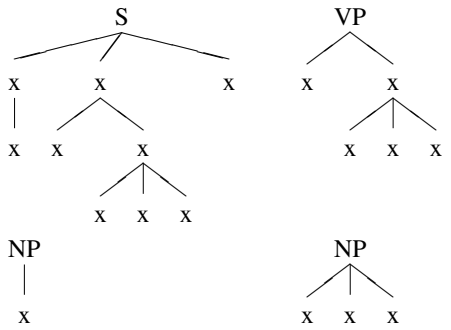
Annotated skeletons are a compromise between rewrite rules and raw skeletons. They hope to form a middle ground between the specificity of rewrite rules and the generality of skeletons. The algorithm to generate these features is almost exactly the same as skeleton's, except the topmost internal node's label is retained (the annotation).

Again, the pseudocode for obtaining annotated skeletons is given below. The function CATEGORY returns the syntactic category of its parameter, *e.g.* VP or CC.

Algorithm 2 Counting annotated skeletons in a parse tree

procedure COUNTANNOTATEDSKELETONS(T)
  $token \leftarrow$ CATEGORY(T) + SKELETON(T)
  INCREMENTCOUNT($token$)
  for each subtree $t \in T$ **do**
    COUNTANNOTATEDSKELETONS(t)
  end for
end procedure

An annotated skeleton feature could be as follows: `(S(())(()(()))())`. Given the example sentence, each subtree above would be given a frequency count of one.



A key difference between the skeleton-based features and the existing rewriting rules features is that the skeleton-based features emphasize pure structural properties by intentionally ignoring the syntactic labels. Thus they can represent text data from an orthogonal perspective to what existing features can capture. Furthermore, an advantage of such features as compared with rewrite rules is that they are generally more frequent, therefore less likely to suffer from data sparseness.

From another perspective, we may also view a skeleton feature as a cluster of syntactically annotated tree structures that share the same underlying structure. An annotated skeleton is simply a more restricted cluster with the root node fixed to a phrasal category.

V. EVALUATION

A. Data Sets

We evaluate the proposed features using three different text categorization tasks that likely benefit from using structural features.

The [22] data set consists of 1008 essays written in English by native Chinese, Japanese, and English students. Essays were classified by their writers' native language. In attempts to keep content uniform, each essay is a response to one of two writing prompts: 1) *It is important for college students to have a part-time job* or 2) *Smoking should be completely banned at all restaurants in the country*. Categorizing text based on assumed nationality would be a useful way to rate one's mastery of a second language. It would also aid in authorship profiling when combined with other methods trained on age and gender.

The [23] data set is 10,686 scored student essays on a range of 0 to 12. Essays were relatively short, all between 150 and 550 words. These essays were originally used as data for a contest in essay scoring. Scores for the essays are an average of three human graders' scores in an attempt to portray the most accurate human judgement.

The data set from [24] consists of 50,000 movie reviews from the International Movie Database, classified as either positive or negative. All movie reviews are scored out of 10, but only clearly negative (score ≤ 4) or clearly positive (score ≥ 7) are included in the data set for data polarization.

B. Measures

To assess classification accuracy for nationality detection and sentiment analysis, we employ the commonly used infor-

Nationality Detection

Method	n_{best}	F_1	A
Word	1	.827	.916
POS	2	.810	.902
FW	2	.728	.876
RR		.778	.844
Skel		.510	.806
ASkel		.721	.870
SC		.703	.844
Branch		.508	.786
Depth		.436	.774
ASkel + Word	**1**	**.885**	**.942**
SC + Word	1	.867	.936
RR + Word	1	.854	.924
ASkel + POS	2	.811	.900
RR + POS	2	.810	.896
Skel + Word	1	.809	.912
ASkel + FW	2	.799	.904
RR + FW	2	.795	.898

Essay Scoring

Method	n_{best}	κ
Word	**2**	**.889**
POS	2	.765
FW	1	.845
RR		.702
Skel		.356
ASkel		.658
SC		.431
Branch		-.087
Depth		-.098
SC + Word	2	.834
ASkel + FW	1	.822
RR + Word	2	.807
ASkel + Word	2	.791
RR + FW	1	.786
ASkel + POS	2	.782
Skel + POS	2	.768
RR + POS	2	.761

Sentiment Analysis

Method	n_{best}	F_1	A
Word	1	.820	.820
POS	3	.662	.662
FW	1	.687	.687
RR		.650	.650
Skel		.556	.557
ASkel		.654	.654
SC		.555	.568
Branch		.493	.499
Depth		.533	.535
Skel + Word	**1**	**.828**	**.828**
RR + Word	1	.824	.824
ASkel + Word	1	.824	.824
SC + Word	1	.822	.822
ASkel + FW	1	.704	.704
RR + FW	1	.686	.686
ASkel + POS	3	.685	.685
Skel + FW	1	.682	.682

Fig. 1. Comparison of single and combined features across data sets. RR, Skel, ASkel, and SC refer to Rewrite Rules, Skeleton, Annotated Skeleton, and Syntactic Category respectively.

mation retrieval measurements F_1 score and accuracy [25].

The essay data set is evaluated differently. As in the original contest, performance is calculated with the quadratic weighted κ metric, described in [26]. In short, the κ metric measures agreement between two raters using a fixed scale, where usually $\kappa \in [0.0, 1.0]$, with 0.0 indicating random agreement and 1.0 indicating exact agreement. For very poor features, it is possible that the score drops below 0.0. The contest's own evaluation script was run on our output.

C. Experiment Design

The main questions we strive to answer are Q_1) Are the new features orthogonal to the existing ones? and Q_2) Can we combine the new features with old ones to improve accuracy?

In order to conduct fair experiments, we created a modular testing framework that easily allows us to exchange features and data sets. Source texts were preprocessed with the Stanford parser [27] and part-of-speech tagger [28], then features were generated based on the preprocessed data. The feature files are then passed to liblinear [29] (an SVM library), where it learns a classifier and performs five-fold cross-validation to evaluate the results. We used the parameter `-s 1` for all runs, referring to *L2-regularized L2-loss support vector classification (dual)*. This configuration has $C = 1, B = -1, \epsilon = 0.1$.

In reference to authorship attribution, [1] notes that the "SVM model is able to avoid overfitting problems even when several thousands of features are used and is considered one of the best solutions of current technology". Hence we chose to use SVMs as our classification method, though of course any classifier could be used.

For word features, 433 stop words based on the Lemur toolkit's [30] stop word list are removed. Then, the words are stemmed according to the Porter2 stemmer [31].

We compare the three baseline features (words, POS tags, function words) with the tree features (rewrite rules, skeletons, annotated skeletons, syntactic categories, tree branch factors, and tree depth).

We partition each data set into two parts; on the first, we perform parameter selection via five-fold cross-validation to find the best n for words, part-of-speech tags, and function words. Then, we select the best-performing n from this set and run it, the tree features, and tree features + best n-grams on the second part, again with five-fold cross-validation.

Software used to run all experiments presented in this paper is open-source and freely available online.[1]

D. Results

Figure 1 shows the evaluation results on the three data sets. Each column is split into three parts; the top records performance for the best-performing single methods for all n-grams. The middle section shows tree-based method results, and the bottom section shows the best-performing combined methods.

On the nationality data set, we see that of the single methods, word unigrams performed the best with an F_1 score of .827, where the best tree feature (RR) had .778. Combining features showed annotated skeletons and unigram words proved most effective ($F_1 = .885$).

The contest that originated the essay data set ended before this work was begun; the first place finisher ended up with a score of $\kappa = .8141$, but this score is not directly comparable to our results since we believe the contest scored entries

[1] https://bitbucket.org/smassung/meta

Nationality Detection F_1 Gain

Method	n	Skel	ASkel	RR	SC
Word	1	-.018	.058	.027	.040
POS	2	-.045	.001	.000	-.025
FW	2	-.074	.071	.067	.024
$\bar{x}$		-.046	**.043**	.031	.013
σ		.028	.037	.034	.033

Essay Scoring κ Gain

Method	n	Skel	ASkel	RR	SC
Word	2	-.140	-.098	-.082	-.055
POS	2	.003	.017	-.004	-.099
FW	1	-.132	-0.23	-.077	-.145
$\bar{x}$		-.090	-.104	-.054	-.100
σ		.080	.123	.044	.045

Sentiment Analysis F_1 Gain

Method	n	Skel	ASkel	RR	SC
Word	1	.008	.004	.004	.002
POS	3	.015	.023	.019	.002
FW	1	-.005	.017	-.001	-.030
$\bar{x}$		.006	**.015**	.007	-.090
σ		.010	.010	.010	.018

Fig. 2. Adding tree features to the best-performing single features changes F_1 and κ scores across the three data sets.

based on a withheld testing set. For this task, word features performed the best, with none of the structural features being beneficial. In fact, both tree depth and tree branch factors did so poorly that their scores ended up negative. This shows that the effectiveness of features clearly depends on the task. Perhaps the essay scoring is more dependent on content rather than writing style.

Similarly to the nationality detection experiment, word unigrams performed the best in the sentiment analysis task ($F_1 = .820$). For combined features, skeleton and unigram words performed the best with ($F_1 = .828$). [24] uses this data set, testing with two folds. They achieved $A = 88.89$. [32] also cites using this data set, with $A = 91.22$ on what we assume to be two folds. We note that our results are significantly less, due to using half the data set for n-gram parameter selection before running the experiments on the other half.

VI. Discussion

A. The Best Features to Combine

Without a doubt, adding additional tree features enhances performance. But which tree features are most effective when combined? We hypothesize that the skeletal features (skeleton and annotated skeleton) result in the best combined performance.

In order to further investigate these hypotheses, Figure 2 shows the relative gain obtained by adding the tree features skeleton, annotated skeleton, rewrite rules, and syntactic categories to the four original methods.

We find that annotated skeleton provides the best performance boost across all three domains. This confirms our suspicions that structural tree information provides the most useful information.

Other interesting results were the performance of non-unigram function words and tree depth. Although previously shown to be a relevant feature in [4], tree depth was one of the worst performers (at least in our data sets).

We do not believe n-grams of function words have seriously been considered as a feature, but bigrams of function words worked well when combined with tree features for the nationality and essay data sets.

B. Validity of Tree Features

We use the correlation coefficient as described in [33] to explore the efficacy of the tree features. Looking at the highest weighted features, we should be able to rationalize their appearance. We choose to leave tree depth and branch factor out of these comparisons since they performed so poorly previously.

Given the following metrics for a term t and a category c_i we can define the probabilities:

1) $P(t, c_i)$: presence of t, membership in c_i
2) $P(t, \overline{c_i})$: presence of t, non-membership in c_i
3) $P(\bar{t}, c_i)$: absence of t, membership in c_i
4) $P(\bar{t}, \overline{c_i})$ absence of t, non-membership in c_i,

Then, with N total documents, the correlation coefficient (CC) can be written as follows:

$$CC(t, c_i) = \frac{\sqrt{N}[P(t, c_i)P(\bar{t}, \overline{c_i}) - P(t, \overline{c_i})P(\bar{t}, c_i))]}{\sqrt{P(t)P(\bar{t})P(c_i)P(\overline{c_i})}}$$

The correlation coefficient can be viewed as a one-sided Chi-square metric [34]. That is, the features selected by CC are most indicative of class membership only (as opposed to membership and non-membership).

The highly-ranked word features are intuitive, especially interesting are "china" and "chines" for c_i = Chinese, and "japan" for c_i = Japanese.

We have a few observations regarding the syntactic features:

1) Somewhat surprisingly, structural tree features are significantly *shorter* for native speakers
2) Native speakers use parenthesis (-RRB- and -LRB-) and colons (but not semicolons) much more than non-native speakers do
3) Non-native speakers more often start phrases with conjunctions
4) Native speakers use more obscure rewrite rules such as VP→MD, ADVP, VP

For example, the sentence *"And if there are unexpected expenses, material for their lesson, for example, they may not be able to pay money to it only with monthly allowance"* shows it beginning with a conjunction, and containing a relatively complex (or convoluted) structure.

c_i = **English**

Words	Skel	ASkel	RR	SC
educ	()	VBG()	NP→(NP)(VP)	VBG
financi	(()()())	DT()	VP→(MD)(ADVP)(VP)	DT
individu	(()()()())	JJ()	NP→(NP)(PP)	:
believ	(()(()()()()))	NNS()	S→(VP)	-RRB-
right	(()())	NP(()()())	PP→(TO)(NP)	-LRB-

c_i = **Chinese**

Words	Skel	ASkel	RR	SC
china	(()(()(()(()(()()))))))	JJR()	ADVP→(DT)	JJR
knowledg	((((()))(()(()))))()()((()))(()(())))	RBR()	ADVP→(DT)(RBR)	RBR
partjob	((()))(()(()))((()))((()))(())))))	ADVP(()()(()(()))(()(())))))	NP→(NP)(,)(SBAR)	$
chines	(()(()()()()))(()(())))	SBAR(()(()(()(()(())))))	ADJP→(JJR)	DT
hold	(()(()(()))(()((()))(()(()(())))))	VP(()(()(()(()(()(())))))	NP→(JJR)	FRAG

c_i = **Japanese**

Words	Skel	ASkel	RR	SC
think	(())	PRP()	NP→(PRP)	PRP
smoke	(()(()))	NP(())	VP→(VBP)(NP)	VBP
seat	((()))(()(())))	VBP()	NP→(NN)	.
money	(()(()))(()(()(()))))())	.()	VP→(VBP)(S)	LS
japan	((()))(()(()))()))	LS()	S→(CC)(NP)(VP)(.)	”

Fig. 3. Samples of the highest ranked features for each language as selected by the correlation coefficient metric. Note that the words are stemmed.

Another sentence by a native speaker *“Indeed, students who have a part-time job (like I did) quickly change their perspective”* shows a non-standard sentence beginning and the (seemingly) popular parentheses.

These observations are intuitive and lend credibility to the tree features, rationalizing their excellent performance when combined with simple features.

VII. Conclusions and Future Work

We compared combinations of simple n-gram text representation models with new and existing tree features. We showed that the novel structural tree features are most effective and when combined with a simpler lexical model, capturing multiple perspectives of the same text.

Using these new methods, we display performance gains on existing corpora across domains. This demonstrates the generality and usefulness of our features. We showed that the new structural features combine better with simple features than existing tree representations such as rewrite rules and tree depth.

Additionally, the structural tree features introduced are not restricted to probabilistic context-free grammars as mainly discussed here; they could be applied to other tree structures as well: abstract syntax trees for source code analysis, dependency parses for more linguistic analysis, and even HTML or XML data for Web page or structured document comparisons.

We aimed to answer Q_1) Are the new features orthogonal to the existing ones? and Q_2) Can we combine the new features with old ones to improve accuracy? Based on our experimental results, we can answer *yes* to both. We assert the new features are orthogonal due to lack of syntactic information and positive F_1 score gain after adding them to the lexical features as shown in Figure 2. We answer Q_2 affirmatively with Figure 1.

In the future we would like to explore these features in tree structures other than PCFGs, as well in other domains such as clustering and information retrieval. Using structural tree features in a topic modeling context would allow distributions of structures to be obtained for each class more easily than with machine learning algorithms such as SVM. This leads to better interpretability of features, offering clearer explanations of why some classes favor certain structures.

We would also like to compare these new methods with the k-embedded-edge subtrees discussed in [6], as well as using their proposed feature reduction frequent tree pattern pruning. Additionally, we would be interested in seeing how the features respond to dimensionality reduction techniques, as the number of skeleton and annotated skeleton features is usually quite large.

Acknowledgments

The authors would like to thank Chase Geigle for his collaborative effort on the toolkit used to run the experiments as well as helpful discussion regarding this work. This material is based upon work supported in part by the National Science Foundation under Grant Number CNS-1027965.

REFERENCES

[1] E. Stamatatos, "A survey of modern authorship attribution methods," *J. Am. Soc. Inf. Sci. Technol.*, vol. 60, no. 3, pp. 538–556, Mar. 2009. [Online]. Available: http://dx.doi.org/10.1002/asi.v60:3

[2] M. Koppel, J. Schler, and S. Argamon, "Computational methods in authorship attribution," *J. Am. Soc. Inf. Sci. Technol.*, vol. 60, no. 1, pp. 9–26, Jan. 2009. [Online]. Available: http://dx.doi.org/10.1002/asi.v60:1

[3] R. H. Baayen, H. van Halteren, A. Neijt, and F. Tweedie, "Outside the cave of shadows: Using syntactic annotation to enhance authorship attribution," *Literary and Linguistic Computing*, vol. 11, no. 3, pp. 121–132, 1996.

[4] M. Chen and K. Zechner, "Computing and evaluating syntactic complexity features for automated scoring of spontaneous non-native speech," in *ACL*, 2011, pp. 722–731.

[5] S. Raghavan, A. Kovashka, and R. Mooney, "Authorship attribution using probabilistic context-free grammars," in *Proceedings of the ACL 2010 Conference Short Papers*, ser. ACLShort '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 38–42. [Online]. Available: http://dl.acm.org/citation.cfm?id=1858842.1858850

[6] S. Kim, H. Kim, T. Weninger, J. Han, and H. D. Kim, "Authorship classification: a discriminative syntactic tree mining approach," in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, ser. SIGIR '11. New York, NY, USA: ACM, 2011, pp. 455–464. [Online]. Available: http://doi.acm.org/10.1145/2009916.2009979

[7] S. Feng, R. Banerjee, and Y. Choi, "Syntactic stylometry for deception detection," in *ACL*, 2012, pp. 171–175.

[8] J. Jiang and C. Zhai, "A systematic exploration of the feature space for relation extraction," in *In Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT07)*, 2007, pp. 113–120.

[9] A. Agarwal, B. Xie, I. Vovsha, O. Rambow, and R. Passonneau, "Sentiment analysis of twitter data," in *Proceedings of the Workshop on Languages in Social Media*, ser. LSM '11. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011, pp. 30–38. [Online]. Available: http://dl.acm.org/citation.cfm?id=2021109.2021114

[10] G. Zhou, L. Qian, and J. Fan, "Tree kernel-based semantic relation extraction with rich syntactic and semantic information," *Inf. Sci.*, vol. 180, no. 8, pp. 1313–1325, Apr. 2010. [Online]. Available: http://dx.doi.org/10.1016/j.ins.2009.12.006

[11] E. Black, S. Eubank, H. Kashioka, D. M. Magerman, R. Garside, and G. Leech, "Beyond skeleton parsing: Producing a comprehensive large-scale general-english treebank with full grammatical analysis," in *COLING*, 1996, pp. 107–112.

[12] R. Wang and G. Neumann, "Recognizing textual entailment using sentence similarity based on dependency tree skeletons," in *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, ser. RTE '07. Stroudsburg, PA, USA: Association for Computational Linguistics, 2007, pp. 36–41.

[13] M. Collins and N. Duffy, "New ranking algorithms for parsing and tagging: kernels over discrete structures, and the voted perceptron," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ser. ACL '02. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002, pp. 263–270. [Online]. Available: http://dx.doi.org/10.3115/1073083.1073128

[14] A. Moschitti, "Making tree kernels practical for natural language learning," in *EACL*, 2006.

[15] T. Kudo and Y. Matsumoto, "A boosting algorithm for classification of semi-structured text," in *EMNLP*, 2004, pp. 301–308.

[16] M. K. Omar and J. W. Pelecanos, "A novel approach to detecting non-native speakers and their native language," in *ICASSP'10*, 2010, pp. 4398–4401.

[17] J. Burstein, K. Kukich, S. Wolff, C. Lu, M. Chodorow, L. C. Braden-Harder, and M. D. Harris, "Automated scoring using a hybrid feature identification technique," in *COLING-ACL'98*, 1998, pp. 206–210.

[18] M. D. Shermis and J. Burstein, *Automated essay scoring: A cross-disciplinary perspective*. MIT Press, 2003, vol. 16.

[19] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Foundations and Trends in Information Retrieval*, vol. 2, no. 1-2, pp. 1–135, 2008.

[20] H. Tang, S. Tan, and X. Cheng, "A survey on sentiment detection of reviews," *Expert Syst. Appl.*, vol. 36, no. 7, pp. 10 760–10 773, Sep. 2009. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2009.02.063

[21] J. Furnkranz, "A study using n-gram features for text categorization," 1998.

[22] S. Ishikawa, "Vocabulary in interlanguage: A study on corpus of english essays written by asian university students (ceeaus)," in *Phraseology, corpus linguistics and lexicography*, ser. Phraseology '09, 2009, pp. 87–100.

[23] T. H. Foundation. (2012, 5) Asap: Automated student essay prize. [Online]. Available: http://www.kaggle.com/c/asap-aes

[24] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, June 2011, pp. 142–150. [Online]. Available: http://www.aclweb.org/anthology/P11-1015

[25] R. Caruana and A. Niculescu-Mizil, "Data mining in metric space: An empirical analysis of supervised learning performance criteria," in *ROCAI'04*, 2004, pp. 9–18.

[26] J. Cohen, "Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit," *Psychological Bulletin*, vol. 70, pp. 213–220, 1968.

[27] D. Klein and C. D. Manning, "Accurate unlexicalized parsing," in *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ser. ACL '03. Stroudsburg, PA, USA: Association for Computational Linguistics, 2003, pp. 423–430. [Online]. Available: http://dx.doi.org/10.3115/1075096.1075150

[28] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, ser. NAACL '03. Stroudsburg, PA, USA: Association for Computational Linguistics, 2003, pp. 173–180. [Online]. Available: http://dx.doi.org/10.3115/1073445.1073478

[29] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *J. Mach. Learn. Res.*, vol. 9, pp. 1871–1874, Jun. 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1390681.1442794

[30] B. Croft and J. Callan. (2012, 10) The lemur project. [Online]. Available: http://www.lemurproject.org

[31] M. Porter. (2012, 10) The english porter2 stemming algorithm. [Online]. Available: http://snowball.tartarus.org/algorithms/english/stemmer.html

[32] S. Wang and C. D. Manning, "Baselines and bigrams: simple, good sentiment and topic classification," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ser. ACL '12. Stroudsburg, PA, USA: Association for Computational Linguistics, 2012, pp. 90–94. [Online]. Available: http://dl.acm.org/citation.cfm?id=2390665.2390688

[33] H. T. Ng, W. B. Goh, and K. L. Low, "Feature selection, perceptron learning, and a usability case study for text categorization," *SIGIR Forum*, vol. 31, no. SI, pp. 67–73, Jul. 1997.

[34] Z. Zheng, X. Wu, and R. Srihari, "Feature selection for text categorization on imbalanced data," *SIGKDD Explor. Newsl.*, vol. 6, no. 1, pp. 80–89, Jun. 2004. [Online]. Available: http://doi.acm.org/10.1145/1007730.1007741