Syntactic Dependency-Based N-grams as Classification Features

Grigori Sidorov[1], Francisco Velasquez[1], Efstathios Stamatatos[2], Alexander Gelbukh[1], and Liliana Chanona-Hernández[3]

[1] Center for Computing Research (CIC), Instituto Politécnico Nacional (IPN), Mexico City, Mexico
[2] University of the Aegean, Greece
[3] ESIME, Instituto Politécnico Nacional (IPN), Mexico City, Mexico
www.cic.ipn.mx/~sidorov

Abstract. In this paper we introduce a concept of syntactic n-grams (sn-grams). Sn-grams differ from traditional n-grams in the manner of what elements are considered neighbors. In case of sn-grams, the neighbors are taken by following syntactic relations in syntactic trees, and not by taking the words as they appear in the text. Dependency trees fit directly into this idea, while in case of constituency trees some simple additional steps should be made. Sn-grams can be applied in any NLP task where traditional n-grams are used. We describe how sn-grams were applied to authorship attribution. SVM classifier for several profile sizes was used. We used as baseline traditional n-grams of words, POS tags and characters. Obtained results are better when applying sn-grams.

Keywords: syntactic n-grams, sn-grams, parsing, classification features, syntactic paths, authorship attribution.

1 Introduction

N-gram based techniques are predominant in modern NLP and its applications. Traditional n-grams are sequences of elements as they appear in texts. These elements can be words, characters, POS tags, or any other elements as they encounter one after another. Common convention is that "n" corresponds to the number of elements in the sequence.

The main idea of this paper is that n-grams can be obtained based on the order in which the elements are presented in syntactic trees. Namely, we follow a path in the tree and construct n-grams, rather than taking them as they appear in the surface representation of the text. Thus, we consider as neighbors the words (or other elements like POS tags, etc.) that follow one another in the path of the syntactic tree, and not in the text. We call such n-grams **syntactic n-grams (sn-grams)**. The great advantage of sn-grams is that they are based on syntactic relations of words and, thus,

I. Batyrshin and M. González Mendoza (Eds.): MICAI 2012, Part II, LNAI 7630, pp. 1–11, 2013.

each word is bound to its "real" neighbors, ignoring the arbitrariness that is introduced by the surface structure.

For example, let us consider two phrases: "*eat with wooden spoon*" vs. "*eat with metallic spoon*", see Fig. 1. Note that we can use both dependency and constituency representations of syntactic relations. They are equivalent if the head of each constituent is known. In our example of constituents, the heads are marked by heavier lines. It is very easy to add information about heads into constituency based grammar, namely, one of the components should be marked as the head in the rules.

In case of dependencies, we follow the path marked by the arrows and obtain sn-grams. In case of constituents we first "promote" the head nodes so that they occupy the places in bifurcations, as shown in Fig. 2. Then we obtain sn-grams starting from the dependent constituencies and taking the heads from the bifurcations.

Let us consider the case of bigrams for this example. If we extract traditional bigrams from the phrases, then they have only one bigram in common: "*eat with*". Meanwhile, if we consider sn-grams, then two common bigrams are found: "*eat with*", "*with spoon*".

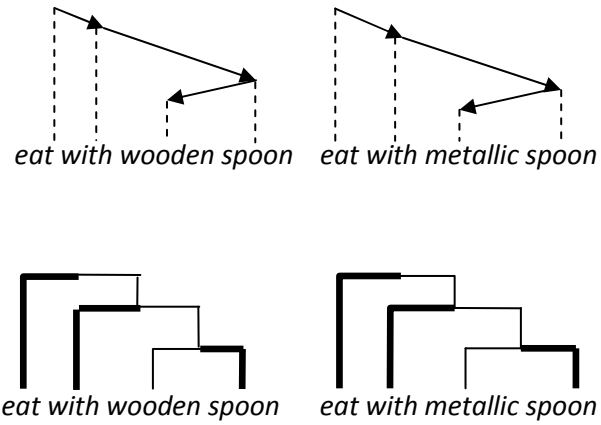

Fig. 1. Representations of syntactic relations

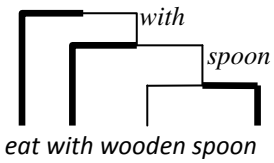

Fig. 2. Promoted head nodes

The same happens in case of trigrams. In case of traditional n-grams, there are no common n-grams, but if we use sn-grams then there is one common trigram: "*eat with spoon*".

In this case, sn-grams allow ignoring the surface phenomenon of the English language that adds an adjective before the noun and in this way spoils traditional bigrams/trigrams. The same happens in case of, say, subordinate clauses, and, in general, in any type of syntactic structures.

Other quite obvious possibility is to construct sn-grams ignoring auxiliary words (stop words). We shall follow the paths in the tree and just pass through the nodes of the stop words. In case of our examples, they have the common sn-bigram "*eat spoon*" that will not be obtained using traditional n-grams.

Sn-grams can be used as classification features in the same manner as traditional n-grams. So, in any task in which we use traditional n-grams, we can apply sn-grams as well.

Obvious problem with sn-grams is that syntactic parsing is required. The parsing can take considerable time and there is a problem of availability of parsers for particular languages, though for well-studied languages like English or Spanish this consideration is not a problem.

In this paper, we apply sn-grams in authorship attribution problem. We conducted experiments that obtain better results for this task with sn-grams than with traditional n-grams. Besides, we believe that sn-grams have real linguistic interpretation as far as the writing style of authors is concerned because they reflect real syntactic relations.

Further in this paper, we first, discuss syntactic n-grams (Section 2), present relevant work for authorship attribution (Section 3), and then present experimental results comparing the proposed approach with traditional methods (Section 4). Finally, we summarize the conclusions drawn by this study.

2 Syntactic n-grams (sn-grams)

The advantage of syntactic n-grams (sn-grams), i.e., n-grams that are constructed using paths in syntactic trees, is that they are less arbitrary than traditional n-grams. Thus, their number is less than the number of traditional n-grams. Besides, they can be interpreted as linguistic phenomenon, while traditional n-grams have no plausible linguistic interpretation—they are merely statistical artifact.

The justification of the idea of sn-grams is related to introduction of linguistic information into statistically based methods. We believe that this idea helps to overcome the main disadvantage of the traditional n-grams—they contain many arbitrary elements, i.e., a lot of noise.

The obvious disadvantage of syntactic n-grams is that previous syntactic processing is necessary. This consumes significant time and it is not easy to apply to some languages, because a syntactic parser and a set of lexical resources that are used by the parser are needed and not for any language these resources are available.

Previously, similar ideas were related to some specific tasks like using additional syntactic information in machine translation [1] or generation in machine translation

[2], without the generalization and taxonomy that we propose in this paper. The term syntactic n-gram is not very common and its importance is underestimated. It is used, for example, in [3] for extraction of polarity of syntactic constituents (chunks) as a whole element.

Note that there are attempts to overcome the disadvantages of traditional n-grams using purely statistical approaches. We should mention skip-grams and Maximal Frequent Sequences (MFS).

Skip-grams are very similar to n-grams, but during their construction some elements of the corresponding sequence are ignored (skipped). It is an attempt as well to avoid possible noise, namely, by considering random variations in texts. There can be skips with various skip steps.

An example of skip-grams: say, for the sequence ABCDE, we obtain traditional bigrams AB, BC, CD, and DE. Skip-bigrams with skip step of 1 will be AC, DB, and CE. Various skip steps can be used. Usually skip-grams also include traditional n-grams, in which case the skip step is zero. The problem with skip-grams is that their number grows very fast.

Maximal Frequent Sequences (MFS) [4] are skip-grams with major frequency, i.e., we take into account only skip-grams whose frequencies are greater than a certain threshold. The problem with MFS is that for their construction complex algorithms should be used and it takes substantial processing time. Another disadvantage of MSF is that unlike sn-grams, they depend on text collection. And similar to skip-grams, no linguistic interpretation of MFS is possible in general case.

According to the **types of elements** that form syntactic n-grams, there can be various types of sn-grams:

- Word sn-grams: the elements of sn-gram are words,
- POS sn-grams: the elements of sn-gram are POS tags,
- Sn-grams of syntactic relations tags: SR tags, the elements of sn-gram are names of syntactic relations,
- Mixed sn-grams: sn-grams are composed by mixed elements like words (lexical units), POS tags and/or SR tags. Since a sn-gram follows a syntactic link, then there are reasons to use mixed sn-grams, for example, they can reflect subcategorization frames. There are many possible combinations regarding to what part—the head word or the dependent word in the relation—should be represented by lexical unit, POS tag, or SR tag. These combinations should be explored experimentally in future.

Note that sn-grams of characters are impossible.

As far as the **treatment of stop words** is concerned, they can be ignored or taken into account, as we mentioned before.

3 Sn-grams of SR Tags

For the experiments described in this paper, we used syntactic relations (SR tags) as elements of sn-grams. Stanford parser was used for determining SR tags, POS tags,

and for construction of dependency-based syntactic trees. Although the parsing process was time consuming for a large corpus, it was performed only once, so the subsequent experiments do not take substantial time.

Let us consider an example sentence "*Economic news have little effect on financial markets*" that is used in some NLP courses. Stanford parser output for dependency relations is:

nn (news-2, Economic-1)
nsubj (have-3, news-2)
root (ROOT-0, have-3)
amod (effect-5, little-4)
dobj (have-3, effect-5)
prep (effect-5, on-6)
amod (markets-8, financial-7)
pobj (on-6, markets-8)

This representation contains the following information: relation name (shown at the beginning of each line) and related words in parenthesis, where the first argument within the parentheses represents the head and the second represents the dependent element. Thus, *amod (effect-5, little-4)* means that there is a relation of adjectival modifier (*amod*) from *effect* to *little*. The numbers correspond to word numbers in the sentence. Stanford parser handles 53 relations.

SR tags are presented in squares over the syntactic tree in Fig. 3.

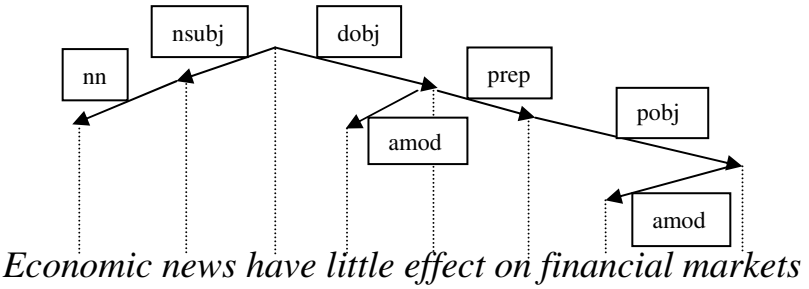

Fig. 3. Dependencies tree of the example

Based on the relations established by the parser, we obtain sn-grams following the arrows. For example, from this tree we can extract the following sn-grams:

nsubj → *nn*
dobj → *amod*
dobj → *prep* → *pobj* → *amod*

Note that we use all possible paths, for example, if we consider trigrams, then the paths *dobj*→*prep*→*pobj* and *prep* → *pobj* → *amod* will be extracted, etc.

4 Relevant Work on Authorship Attribution

Authorship attribution deals with an old and difficult question: how to assign a text of unknown or disputed authorship to a member of a set of candidate authors for whom undisputed samples of texts are available [8]. Despite its application to literary works, the rapid expansion of online text in Internet media (e.g., blogs, e-mail messages, social media postings, etc.) revealed practical applications of authorship attribution usually associated with forensic tasks [11].

The automated approaches to this problem involve the use of statistical or machine learning methods [6]. From the machine learning point of view, authorship attribution can be seen as a single-label multi-class classification task [10]. There are two basic steps: first, the texts should be appropriately represented as vectors of numerical values and, then, a classification algorithm can use these vectors to estimate the probability of class membership for any given text.

Thousands of stylometric features have been proposed so far. These features can be distinguished in the following main categories according to the textual analysis they require [6]: lexical features (e.g., function word frequencies, word n-gram frequencies, vocabulary richness measures, etc.), character features (e.g., letter frequencies and character n-grams), syntactic features (e.g., part-of-speech tag frequencies, sentence and phrase structure measures, rewrite rule frequencies etc.), semantic features (e.g., synonym measures, semantic dependency measures etc.), and application-specific features (e.g., font size, font color, specific word frequencies, etc.). Until now, a number of studies [13, 14, 18] have shown that the most effective measures are lexical and character features.

Note that information about syntactic relations usually is not used in this task, being one of the exceptions [1], where the authors analyze syntactic rewriting rules. In this paper we show that syntactic relations taken as sn-grams represent very effective measure.

In this paper, we use as baseline methods character features, lexical features (words), and POS tags obtained using traditional n-grams, i.e., according to the appearance of the elements in texts.

5 Experimental Results and Discussion

The experiments were performed over the corpus data. The corpus used in our study includes texts downloaded from the Project Gutenberg. We selected books of native English speaking authors that have literary production in a similar period. In this paper, all experiments are conducted for the corpus of 39 documents by three authors.

For evaluation of the experiments, we used 60% of the data for training, and the rest 40% for classification, as presented in Table 1.

We used WEKA implementation of Support Vector Machines (SVM) classifier. SVM is known to produce very good results in the task of authorship attribution.

We used several features as baseline: word based features, character based features, and POS tags. For baseline features, we used traditional n-gram technique, i.e., the elements are taken as they appear in the texts. We applied sn-grams technique for the same corpus with the results that outperform baseline methods.

Table 1. Training and classification data

	Training		Classification	
Author	Novels	Size (MB)	Novels	Size (MB)
Booth Tarkington	8	3.6	5	1.8
George Vaizey	8	3.8	5	2.1
Louis Tracy	8	3.6	5	2.2
Total	**24**	**11**	**15**	**6.1**

We use the term "profile size" for representing the first most frequent n-grams/sn-grams, e.g., for profile size of 400 only first 400 most frequent n-grams are used. We tested various thresholds for profile size and selected five thresholds for profile size as presented in all tables with the results.

When the table cell contains *NA (not available)*, it means that our data were insufficient to obtain the corresponding number of n-grams. It happens only with bigrams because in general there are less bigrams than trigrams, etc. In these cases the total number of all bigrams is less than the given profile size.

In Tables 2, 3 and 4 the results of the baseline methods are presented for the selected baseline methods. Table 5 contains the results obtained using sn-grams. For better appreciation of the comparison of the results, we present Tables 6 to 9, where the results are grouped by the size of n-grams/sn-grams.

Table 2. Word based n-grams (baseline)

Profile size	**n-gram size**			
	2	**3**	**4**	**5**
400	86%	81%	67%	45%
1,000	86%	71%	71%	48%
4,000	86%	95%	67%	48%
7,000	86%	90%	71%	45%
11,000	89%	90%	75%	33%

Table 3. Character based n-grams (baseline)

Profile size	**n-gram size**			
	2	**3**	**4**	**5**
400	90%	76%	81%	81%
1,000	95%	86%	86%	76%
4,000	90%	95%	90%	86%
7,000	NA	90%	86%	86%
11,000	NA	100%	90%	86%

Table 4. n-grams based on POS tags (baseline)

Profile size	n-gram size			
	2	**3**	**4**	**5**
400	90%	90%	76%	62%
1,000	95%	90%	86%	67%
4,000	NA	100%	86%	86%
7,000	NA	100%	90%	86%
11,000	NA	95%	90%	86%

Table 5. sn-grams based on SR tags

Profile size	n-gram size			
	2	**3**	**4**	**5**
400	100%	100%	87%	93%
1,000	100%	100%	87%	93%
4,000	100%	100%	93%	73%
7,000	100%	100%	87%	87%
11,000	100%	100%	93%	87%

In the following tables we present the data grouped by n-gram sizes.

Table 6. Comparison for bigrams

Profile size	Features			
	sn-grams of SR tags	**n-grams of POS tags**	**Character based n-grams**	**Word based n-grams**
400	100%	90%	90%	86%
1,000	100%	95%	95%	86%
4,000	100%	NA	90%	86%
7,000	100%	NA	NA	86%
11,000	100%	NA	NA	89%

Table 7. Comparison for trigrams

Profile size	Features			
	sn-grams of SR tags	**n-grams of POS tags**	**Character based n-grams**	**Word based n-grams**
400	100%	90%	76%	81%
1,000	100%	90%	86%	71%
4,000	100%	100%	95%	95%
7,000	100%	100%	90%	90%
11,000	100%	95%	100%	90%

Table 8. Comparison for 4-grams

Profile size	Features			
	sn-grams of SR tags	**n-grams of POS tags**	**Character based n-grams**	**Word based n-grams**
400	87%	76%	81%	67%
1,000	87%	86%	86%	71%
4,000	93%	86%	90%	67%
7,000	87%	90%	86%	71%
11,000	93%	90%	90%	75%

Table 9. Comparison for 5-grams

Profile size	Features			
	sn-grams of SR tags	**n-grams of POS tags**	**Character based n-grams**	**Word based n-grams**
400	93%	62%	81%	45%
1,000	93%	67%	76%	48%
4,000	73%	86%	86%	48%
7,000	87%	86%	86%	45%
11,000	87%	86%	86%	33%

It can be appreciated that in all cases sn-gram technique outperforms the technique based on traditional n-grams. We consider that SR tags and POS tags are similar for the purposes of our comparison; both are small sets of tags: 36 and 53 elements, associated with words. Note that the majority of these elements have very low frequency.

As can be seen, topline of our task is very high: 100%. It is related to the fact that we use much data and our classification only distinguishes between three classes. In some cases, baseline methods also reach the topline. Still, it happens only for a small number of specific profile sizes. The best results are obtained by sn-grams using bigrams and trigrams for any profile size. For any combination of parameters baseline methods got worse results than sn-grams.

The question can arise if it is worth working with the small number of classes. In our opinion, it is useful and important. First of all, authorship attribution is often a real world application in case of a dispute over the authorship of a document, and in this case the number of classes is reduced to two or three, i.e., it is our situation.

6 Conclusions

In this paper we introduce a concept of syntactic n-grams (sn-grams). The difference between traditional n-grams and sn-grams is related to the manner of what elements are considered neighbors. In case of sn-grams, the neighbors are taken by following syntactic relations in syntactic trees, while traditional n-grams are formed as they appear in texts.

Any syntactic representation can be used for application of sn-gram technqiue: dependency trees or constituency trees. In case of dependency tree, we should follow the syntactic links and obtain sn-grams. In case of constituency trees, some additional steps should be made, but these steps are very simple.

Sn-grams can be applied in any NLP task where traditional n-grams are used.

We conducted experiments for authorship attribution task using SVM for several profile sizes. Relatively large corpus of works of three authors was used.

We used as baseline traditional n-grams of words, POS tags and characters. The results show that sn-gram technique outperforms the baseline technique.

As future work, we will apply the idea of sn-grams to other NLP tasks. As far as our particular data is concerned, we will perform the exhaustive comparison of all features using technique of sn-grams. We also plan to experiment with various types of mixed sn-grams.

Acknowledgements. Work done under partial support of Mexican government (projects CONACYT 50206-H and 83270, SNI) and Instituto Politécnico Nacional, Mexico (projects SIP 20111146, 20113295, 20120418, COFAA, PIFI), Mexico City government (ICYT-DF project PICCO10-120) and FP7-PEOPLE-2010-IRSES: Web Information Quality - Evaluation Initiative (WIQ-EI) European Commission project 269180. We also thank Sabino Miranda and Francisco Viveros for their valuable and motivational comments.

References

1. Khalilov, M., Fonollosa, J.A.R.: N-gram-based Statistical Machine Translation versus Syntax Augmented Machine Translation: comparison and system combination. In: Proceedings of the 12th Conference of the European Chapter of the ACL, pp. 424–432 (2009)
2. Habash, N.: The Use of a Structural N-gram Language Model in Generation-Heavy Hybrid Machine Translation. In: Belz, A., Evans, R., Piwek, P. (eds.) INLG 2004. LNCS (LNAI), vol. 3123, pp. 61–69. Springer, Heidelberg (2004)
3. Agarwal, A., Biads, F., Mckeown, K.R.: Contextual Phrase-Level Polarity Analysis using Lexical Affect Scoring and Syntactic N-grams. In: Proceedings of the 12th Conference of the European Chapter of the ACL (EACL), pp. 24–32 (2009)
4. García-Hernández, R.A., Martínez Trinidad, J.F., Carrasco-Ochoa, J.A.: Finding Maximal Sequential Patterns in Text Document Collections and Single Documents. Informatica 34(1), 93–101 (2010)
5. Baayen, H., Tweedie, F., Halteren, H.: Outside The Cave of Shadows: Using Syntactic Annotation to Enhance Authorship Attribution. Literary and Linguistic Computing, 121–131 (1996)
6. Stamatatos, E.: A survey of modern authorship attribution methods. Journal of the American Society for information Science and Technology 60(3), 538–556 (2009)
7. Holmes, D.I.: The evolution of stylometry in humanities scholarship. Literary and Linguistic Computing 13(3), 111–117 (1998)
8. Juola, P.: Authorship Attribution. Foundations and Trends in Information Retrieval 1(3), 233–334 (2006)
9. Juola, P.: Ad-hoc authorship attribution competition. In: Proceedings of the Joint Conference of the Association for Computers and the Humanities and the Association for Literary and Linguistic Computing, pp. 175–176 (2004)
10. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys 34(1) (2002)
11. Abbasi, A., Chen, H.: Applying authorship analysis to extremist-group web forum messages. IEEE Intelligent Systems 20(5), 67–75 (2005)
12. van Halteren, H.: Author verification by linguistic profiling: An exploration of the parameter space. ACM Transactions on Speech and Language Processing 4(1), 1–17 (2007)
13. Grieve, J.: Quantitative authorship attribution: An evaluation of techniques. Literary and Linguistic Computing 22(3), 251–270 (2007)
14. Luyckx, K.: Scalability Issues in Authorship Attribution. Ph.D. Thesis, University of Antwerp (2010)
15. Argamon, S., Juola, P.: Overview of the international authorship identification competition at PAN-2011. In: 5th Int. Workshop on Uncovering Plagiarism, Authorship, and Social Software Misuse (2011)
16. Diederich, J., Kindermann, J., et al.: Authorship attribution with support vector machines. Applied Intelligence 19(1), 109–123 (2003)
17. Escalante, H., Solorio, T., et al.: Local histograms of character n-grams for authorship attribution. In: 49th Annual Meeting of the Association for Computational Linguistics, pp. 288–298 (2011)
18. Keselj, V., Peng, F., et al.: N-gram-based author profiles for authorship attribution. Computational Linguistics 3, 225–264 (2003)
19. Koppel, M., Schler, J., et al.: Authorship attribution in the wild. Language Resources and Evaluation 45(1), 83–94 (2011)
20. Koppel, M., Schler, J., et al.: Measuring differentiability: unmasking pseudonymous authors. Journal of Machine Learning Research, 1261–1276 (2007)