Improving Corpus Annotation Quality Using Word Embedding Models

Attila Novák

Abstract—**Web-crawled corpora contain a significant amount of noise. Automatic corpus annotation tools introduce even more noise performing erroneous language identification or encoding detection, introducing tokenization and lemmatization errors and adding erroneous tags or analyses to the original words. Our goal with the methods presented in this article was to use word embedding models to reveal such errors and to provide correction procedures. The evaluation focuses on analyzing and validating noun compounds identifying bogus compound analyses, recognizing and concatenating fragmented words, detecting erroneously encoded text, restoring accents and handling the combination of these errors in a Hungarian web-crawled corpus.**

Index Terms—**Corpus linguistics, lexical resources, corpus annotation, word embeddings.**

I. Introduction

STATISTICAL methods of natural language processing rely on large written text corpora. The main source of such texts is the web, where the amount of user-generated and social media contents are increasing rapidly. This phenomena and the structure of web contents and their production strategies result in large, but often very noisy text collections. These corpora are not only full of non-standard word forms, but also HTML entities, encoding errors, and deficient written language use (such as the complete lack of accents in texts written in a language with an accented writing system). Thus, even simple preprocessing tools, for example a custom tokenizer, might fail to process these texts correctly. One solution to this problem would be the adaptation of these tools to these specific cases and their careful application to such noisy texts. However, it is more often the case that web-crawled texts are collected in large quantities, sometimes for several different languages in parallel, and there is no time or satisfactory language competence to tune these general preprocessing tools. Moreover, if these texts are to be analyzed more deeply, the errors propagate through the whole processing chain, and become uncontrollable. The other solution to this problem is to apply a postcorrection method that is able to detect and if possible, correct these errors due to the nature of the source of the texts.

In this paper, we propose a method for discovering and categorizing corpus annotation errors. We were provided with a large web-crawled corpus in Hungarian. However, its quality fell short of expectations. Since the size of the corpus was about 3 billion tokens, the errors became apparent only when it was used for a certain task, i.e. building word embedding models. This lead to the idea to use these models to detect and correct erroneous parts of the corpus. We have finally built a corpus cleaning chain handling different deficiencies of the corpus.

The paper is structured as follows. First, a brief introduction to related problems and to word embedding models is presented, which were used in several further processing steps. Then, in Section IV, our method for detecting and correcting tokenization problems is described. This step is followed by a method for detecting and restoring accents in unaccented portions of the corpus. In Section VI, we propose a method for identifying bogus noun compound analyses to eliminate annotation errors introduced by morphological analysis.

II. Related Work

The main source of noise in a web-crawled corpus is the collection of texts from social media sites. These types of texts form an independent area of research. Regarding normalization, a couple of studies aim at normalizing non-standard word usage in these texts trying to compare and convert them to wordforms corresponding to orthographic standards [1], [2] . Most recent methods published for such tasks use word embedding space transformations to estimate the normalized form of a non-standard word by finding the nearest wordform in the space of standard words to the transformed vector from the space of non-standard words [3].

Another type of problem we try to handle is the correct analysis of noun compounds. This problem is independent from the quality of the original corpus, it is present at the level of morphological analysis. In Hungarian, similarly to German, noun compounds can be built by combining nearly any nouns. However, some wordforms can be misleading, because inflected forms or even lemmas may have a grammatically possible but nonsensical compound analysis. The morphological analyzer is not able to decide in such cases what the correct decomposition of such forms are, except when the compound is explicitly included in the lexicon of the analyzer. The corpus contains a great amount of composite word forms to which the analyzer applies

Manuscript received on March 18, 2016, accepted for publication on June 17, 2016, published on June 25, 2016.

Attila Novák is with MTA-PPKE Hungarian Language Technology Research Group, Pázmány Péter Catholic University, Faculty of Information Technology and Bionics, 50/a Práter street, 1083 Budapest, Hungary (e-mail: novak.attila@itk.ppke.hu).

productive compound analysis. Dima et al. [4] have proposed a method for interpreting noun compounds for English using word embeddings. However, neither the inflectional ambiguity nor the segmentation problem are present in their case, only the task of interpretation is addressed. Checking the validity of single word compounds is similar to that of detecting multiword expressions and exploring their compositionality [5]. These studies, however, aim at determining the level of compositionality in the already identified multiword expressions. These approaches do not deal with ambiguity and the detection of real and unreal compounds. Nevertheless, we also relied on compositionality measures to evaluate possible compound analyses in the algorithm presented in this paper.

III. Word embedding models

We built two types of models using the `word2vec`[1] tool, a widely-used framework for creating word embedding representations [6], [7]. This tool implements both skip-gram and CBOW (continuous bag of word) models, generally used for building the embedding vectors. As the CBOW model has proved to be more efficient for large training corpora, we used this model. In our models, the radius of the context window used at the training phase of the model was set to 5 and the number of dimensions to 300.

Then we applied different types of preprocessing to the corpus in order to adapt the method to the agglutinating behavior of Hungarian (or to any other morphologically rich language having a morphological analyzer/tagger at hand). First, we built a model from the tokenized but otherwise raw corpus. This model assigns different vectors to different suffixed forms of the same lemma, while homonymous word forms share a single vector representation. In the other model we used a part-of-speech tagged and disambiguated version of the corpus. This was done using the PurePos part-of-speech tagger [8], which utilizes the Humor morphological analyzer [9], [10], [11] for morphological analysis and also performs lemmatization. We built a model in which each word in the corpus is represented by the sequence of two tokens: one consisting of the lemma and the other of the morphological tags generated by the tagger/morphological analyzer like in [12].

When using the models built from the raw and the annotated corpus for other tasks, different types of errors were revealed when investigating lists of similar words for certain seed terms. These were the following: (1) simple misspellings, (2) unaccented forms, (3) forms with character encoding errors, (3) word fragments, (4) annotation errors. Even though these erroneous forms were to some extent also semantically related, the errors often overshadowed semantic similarity and words with the same error type were clustered by the models.

Performing deeper analysis regarding the source of these errors lead us to the inadequate quality of the original corpus. However, we could rely on the embedding models created from the erroneous corpus to implement methods aiming at improving the quality of the corpus and its annotation.

IV. Detecting word fragments and tokenization errors

One of the problems the models revealed was the presence of a great number of word fragments. These were for the most part introduced by the custom tokenizer applied to the texts. Fortunately, the tokenizer inserted a glue tag `<g/>` at places where it broke single words. The glue tag indicates that there was no whitespace at the position of the tag in the original text. Examples for such situations are hyphens or other punctuation marks, numbers within words, or HTML entities within words. However, some of these splits were nonsense, for example if an HTML entity within a word indicated possible hyphenation boundaries, but not real segmentation, then these words were split at all such boundaries. The tokenizer also segmented words at HTML entities encoding simple letter characters. Fortunately, these erroneous splits could be undone by finding glue tags in contexts where they should not occur.

However, not all word splits were explicitly marked with these tags. If an HTML tag was inserted into a word, then the word was simply split at these points, leaving no track of its original form. These could only be tracked by finding the original HTML source of the texts.

Another source of seemingly fragmented words was due to incorrect lemmatization. These forms appeared only in the model built from the analyzed corpus and could be identified by looking them up in the embedding model of surface forms. If a fragment appeared only in the analyzed model, then it was a lemmatization error.

In order to measure the relevance of this error in the corpus, and the proportion of the various causes, we collected a set of word fragments from the corpus. This could be done easily by querying the embedding models for the nearest neighbors of some fragments. Such queries resulted in lists of fragments hardly containing any real words. These real words could then be easily filtered out by clustering the resulting set. The hierarchical clustering algorithm we applied, grouped real words into a few distinct clusters. Projecting this initial set of fragments to the whole vocabulary revealed that 3.7% (!) of the most frequent 100,000 noun, adjective and verb “lemmas” identified by the annotation system was due to the presence of such fragments in the corpus.

Revisiting the glue tags introduced by the tokenizer and unifying those words that should not have been split at this stage corrected 49.36% of these errors. Then, since these fragments were collected from the analyzed model (due to its more robust and coherent representation of words), fragments in the remaining list were checked in the embedding model of surface forms in order to eliminate the errors introduced by the lemmatizer. This method revealed that 17.08% of the original list originated here. This result can be a good starting point for improving the accuracy of the lemmatizer in PurePos

[1] https://code.google.com/p/word2vec/

TABLE I
SUMMARY OF RESULTS REGARDING WORD FRAGMENTS IN THE CORPUS

type	percentage of tokens
original state	3.70%
corrected state	1.04%
type	**percentage of word fragments**
incorrect splits indicated by glue tags	49.36%
lemmatization errors	17.08%
validated by a spell checker	5.45%

used for words not analyzed by the morphological analyzer. However, handling that problem is out of the scope of this paper. Since a major part of these lemmatization errors is due to spelling or capitalization errors in the original corpus, which resulted in the failure of morphological analysis and the lemmatizer guesser being applied, most of these errors should be handled by identifying and correcting the errors in the corpus. A further 5.45% of the list was validated by a spellchecker as correct word form. However, this did not mean that these strings could not be fragments of longer words at the same time. Thus, 71.89% of these fragmentation errors could be eliminated, reducing the original percentage of 3.7% to 1.04%. These remaining errors are mostly due to the incorrect parsing of the HTML source, splitting words in case of HTML tags within words, without leaving any trace of doing this. Since we had no access to the original HTML source of the corpus, we could not correct these errors. Table I summarizes the results.

V. RESTORING ACCENTS

In Hungarian, umlauts and acute accents are used as diacritics for vowels. Acute accents mark long vowels, while umlauts are used to indicate the frontness of rounded vowels *o*→*ö* [o→ø] and *u*→*ü* [u→y], like in German. A combination of acutes and umlauts is the double acute diacritic to mark long front rounded vowels *ő* [øː] and *ű* [yː]. Long vowels generally have essentially the same quality as their short counterpart (*i-í, ü-ű, u-ú, ö-ő, o-ó*). The long pairs of the low vowels *a* [ɔ] and *e* [ɛ], on the other hand, also differ in quality: *á* [aː] and *é* [eː]. There are a few lexicalized cases where there is a free variation of vowel length without distinguishing meaning, e.g. *hova*∼*hová* 'where to'. In most cases, however, the meaning of differently accented variants of a word is quite different. Table II shows all the possible unaccented-accented pairs of vowels in Hungarian together with their distribution in a corpus of 1 804 252 tokens.

TABLE II
POSSIBLE ACCENT VARIATIONS IN HUNGARIAN

a	a: 70.33%; á: 29.66%
e	e: 73.40% é: 26.59%
i	i: 86.04% í: 13.95%
o	o: 55.41% ó: 14.65% ö: 15.82% ő: 14.10%
u	u: 46.96% ú: 12.72% ü: 29.98% ű: 10.32%

TABLE III
RATIO OF EACH LANGUAGE/TEXT TYPE IN THE CORPUS

Language/type	Number of words	Percentage
All	2684584137	100.00%
Hungarian	2560265742	95.37%
Encoding error	88668867	3.30%
Unaccented Hungarian	9177770	**0.34%**
English	7535446	0.28%
German	4202044	0.16%
French	767515	0.03%
Latin etc.	1311286	0.05%
Short rest	12655467	0.47%

Due to their meaning distinguishing function, it is crucial for any further processing steps to have the accents in the texts. However, due to the widespread use of smart mobile devices, more and more texts on the web are created without accents, because these devices do not really provide a comfortable and fast possibility to type accented characters. The embedding models used in our experiments also justified this assumption, generating unaccented forms as nearest neighbors for some seed words. In order to detect such portions of the corpus, we trained the TextCat language guesser [13] on standard and unaccented Hungarian. We also used language models for other languages we identified as being present in the corpus with this tool to categorize each paragraph of the original corpus. Furthermore, two more categories were also considered, namely encoding errors and short paragraphs (the language of which cannot be reliably identified by TextCat). Erroneous identification of the source code page of HTML pages resulted in encoding errors, which often also resulted in fragmentation of words by the tokenizer. Even though compared to the size of the whole corpus, the amount of text written in other languages, missing accents or being erroneously decoded does not seem to be too much, errors of this type affect the vocabulary present in the corpus significantly because even an erroneous subcorpus of a size of a couple of 10 million tokens results in a million of erroneous word types injected into the vocabulary.

The results are shown in Table III.

As it can be seen from the percentages, the ratio of texts containing encoding errors and unaccented texts is quite high.

Once recognized, unaccented paragraphs can be corrected by applying an accent restoration system. We used the one described in [14], a system based on an SMT decoder augmented with a Hungarian morphological analyzer. Since we had access to that system and to the model built for the experiments described in the paper, we did not have to train, but could just use the system as it was. This tool could restore accented words with an accuracy above 98%.

VI. CORRECTING BOGUS NOUN COMPOUND ANALYSES

In Hungarian, noun compounds are very frequent. The most productive compounding pattern is concatenating nouns. In many cases certain inflected forms can also be analyzed as a compound. In such cases the morphological analyzer is not able to choose the correct segmentation unless the compound

TABLE IV
EXAMPLES FOR AMBIGUOUS SEGMENTATIONS OF WORDS. THE CORRECT (OR MORE PROBABLE) ONES ARE TYPESET IN BOLDFACE.

original form	possible segmentation	meaning in English
gázló	gáz+ló N	'gas+horse'
	gázló N	'ford'
tűnő	tű+nő N	'needle+woman'
	tűnő V.PrtPres	'looking like sg.'

is explicitly included in the lexicon of the analyzer. Some examples for ambiguous segmentation are shown in Table IV

Although the lexicon of the morphological analyzer contains many compound stems, nevertheless in a big corpus there will always be words where productive compounding is needed to yield a valid analysis. Moreover, although many bogus compound analyses are prevented in the analyzer by excluding certain nouns from compounding, productive compounding may still result in bogus compound analyses. Thus, handling this very elemental problem can also be considered a corpus quality issue, because morphological analysis is the basis of many other NLP tasks. And again, we used word embedding models to create a method for identifying erroneous compound segmentation. The morphological analyzer used in our experiments [9], [10] is able to suggest the various possible segmentations, but is not able to choose the correct one. The problem to be solved can be considered a binary classification problem, in which the algorithm has to decide whether a segmentation candidate is correct or not.

First, all words from the corpus were selected for which the morphological analyzer suggested at least one productively generated compound segmentation (either correct or incorrect). From this list of 6,122,267 words, a random selection of 1300 words were taken apart for development and testing purposes. This set was manually checked and the correct segmentations were chosen.

We created one baseline system that queried all possible compound members for all analyses returned by the morphological analyzer, and sorted them by their similarity to the original word form in the vector space model generated from the raw corpus. For compounds consisting of more than two elements, all compound member sequences that did not match the whole stem were also included in the list. We then selected the top-ranked item in this list (the one closest in the vector space to the original word form), and excluded all analyses which were not compatible with this item. If the top-ranked item matched a lexically given segmentation in the lexicon of the morphological analyzer, we accepted that segmentation. All analyses not excluded by the top-ranked item were kept as possible ones.

In the other system, several features were determined for each word for each segmentation suggested by the analyzer. First, the constructing elements of the actual segmentation were ranked according to their similarity to the original form, for which the similarity values were extracted from the embedding model (this step corresponds to the first baseline system). In addition, assuming that the meaning of compounds should be compositional, the 10 nearest neighbors for each element were also retrieved from the embedding model, and all of these were combined using the segmentaton of the original word as a model, producing analogous variants for the original word where compound members are substituted with synonymous words. This list of analogous words was then also ordered by each item's similarity to the original word. Having these ordered lists, the following numerical features were derived:

- A: The similarity of the first-ranked element of the original segmentation
- B: The average of the similarities of all elements of the original segmentation
- C: The similarity of the first-ranked analogous variant (or zero, of no analogous variant was found)
- D0: The length of the list of analogous variants with similarity greater than zero
- D1: The average of the similarities of analogous variants with similarity greater than zero
- D2: D0*D1
- D3: The average of D0, D1 and C

Once these features were extracted, a simple binary decision tree was trained for each of these features individually and for the combination of all of these features. For training and testing, we applied a 10-fold crossvalidation using the previously separated and manually labelled list of 1200 words with a different 9:1 split in each run. The results are shown in Table V. The table contains the accuracy of each system, i.e. the ratio of correctly predicting the correctness of a given segmentation for a certain word. As it can be seen from the table, the most significant feature turned out to be the length of the list of analogous variants. This suggests, that if there is a large enough number of words created by substituting each element of a proposed segmentation with words of similar meaning and the resulting compositions are existing words, then the segmentation can be considered as a valid compound with almost 90% certainty. While the first baseline system relied on lexical knowledge embodied in the compound analyses listed in the lexicon of the morphological analyzer, the decision-tree-based systems did not use that information. The success of the D0 system seems to indicate that compositionality and variability is an important property of productive compounding.

Thus, integrating this feature into the compounding model implemented in the morphological analyzer can also have a beneficial effect on the quality of the annotation.

VII. CONCLUSION

In this paper we explored methods based on continuous vector space models that can be used to identify and correct errors in corpus annotation ranging from errors resulting from erroneous language identification or encoding detection through tokenization and lemmatization errors to erroneous

TABLE V
THE PRECISION OF EACH SYSTEM CREATED FOR VALIDATING CORRECT SEGMENTATIONS OF POSSIBLE COMPOUNDS

System	Precision
first baseline	86.45%
decision tree for feature A	82.32%
decision tree for feature B	82.41%
decision tree for feature C	85.17%
decision tree for feature D0	**90.34%**
decision tree for feature D1	85.43%
decision tree for feature D2	84.22%
decision tree for feature D3	85.43%
decision tree for all features	85.34%

compound analyses. As these models effectively map tokens having a similar distribution to similar locations in vector space, they can be used to retrieve and cluster tokens in the corpus that are there due to the same types of errors in the annotation tool chain revealing the nature and the possible source of these error. Moreover, the distributional models can also be used to identify possible errors in the annotation such as bogus compound analyses exploiting the fact that productive compounding is in general a compositional operation. Here we did not explore the possibility of taking advantage of these models for the identification and correction of errors inherently present in the corpus, such as spelling errors. Nevertheless, that seems to be another promising application area.

REFERENCES

[1] V. K. Rangarajan Sridhar, "Unsupervised text normalization using distributed representations of words and phrases," in *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*. Denver, Colorado: Association for Computational Linguistics, June 2015, pp. 8–16.

[2] C. Li and Y. Liu, "Improving text normalization via unsupervised model and discriminative reranking," in *Proceedings of the ACL 2014 Student Research Workshop*. Baltimore, Maryland, USA: Association for Computational Linguistics, June 2014, pp. 86–93.

[3] L. Tan, H. Zhang, C. Clarke, and M. Smucker, "Lexical comparison between Wikipedia and Twitter corpora by using word embeddings," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Beijing, China: Association for Computational Linguistics, July 2015, pp. 657–661. [Online]. Available: http://www.aclweb.org/anthology/P15-2108

[4] C. Dima and E. Hinrichs, "Automatic noun compound interpretation using deep neural networks and word embeddings," in *Proceedings of the 11th International Conference on Computational Semantics*. London, UK: Association for Computational Linguistics, April 2015, pp. 173–183.

[5] B. Salehi, P. Cook, and T. Baldwin, "A word embedding approach to predicting the compositionality of multiword expressions," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Denver, Colorado: Association for Computational Linguistics, May–June 2015, pp. 977–983. [Online]. Available: http://www.aclweb.org/anthology/N15-1099

[6] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 3111–3119.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *CoRR*, vol. abs/1301.3781, 2013.

[8] G. Orosz and A. Novák, "PurePos 2.0: A hybrid tool for morphological disambiguation," in *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2013)*. Hissar, Bulgaria: INCOMA Ltd. Shoumen, BULGARIA, 2013, pp. 539–545.

[9] A. Novák, "Milyen a jó humor? [What is good humor like?]," in *I. Magyar Számítógépes Nyelvészeti Konferencia [First Hungarian conference on computational linguistics]*. Szeged: SZTE, 2003, pp. 138–144.

[10] G. Prószéky and B. Kis, "A unification-based approach to morpho-syntactic parsing of agglutinative and other (highly) inflectional languages," in *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, ser. ACL'99. Stroudsburg, PA, USA: Association for Computational Linguistics, 1999, pp. 261–268.

[11] A. Novák, "A new form of humor – mapping constraint-based computational morphologies to a finite-state representation," in *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, N. Calzolari, K. Choukri, T. Declerck, H. Loftsson, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, and S. Piperidis, Eds. Reykjavik, Iceland: European Language Resources Association (ELRA), May 2014, pp. 1068–1073.

[12] B. Siklósi, "Using embedding models for lexical categorization in morphologically rich languages," in *Computational Linguistics and Intelligent Text Processing: 17th International Conference, CICLing 2016, Konya, Turkey, April 3-9, 2016*, A. Gelbukh, Ed. Cham: Springer International Publishing, 2016.

[13] K. Hornik, P. Mair, J. Rauch, W. Geiger, C. Buchta, and I. Feinerer, "The textcat package for n-gram based text categorization in R," *Journal of Statistical Software*, vol. 52, no. 6, pp. 1–17, 2013.

[14] A. Novák and B. Siklósi, "Automatic diacritics restoration for hungarian," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, September 2015, pp. 2286–2291. [Online]. Available: http://aclweb.org/anthology/D15-1275