Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), pages 4798–4801
Marseille, 11–16 May 2020

CBOW-tag: a Modified CBOW Algorithm for Generating Embedding Models from Annotated Corpora

Attila Novák, László János Laki, Borbála Novák
MTA-PPKE Hungarian Language Technology Research Group,
Pázmány Péter Catholic University, Faculty of Information Technology and Bionics
Práter u. 50/a, 1083 Budapest, Hungary
{surname.firstname}@itk.ppke.hu

Abstract

In this paper, we present a modified version of the CBOW algorithm implemented in the fastText framework. Our modified algorithm, CBOW-tag builds a vector space model that includes the representation of the original word forms and their annotation at the same time. We illustrate the results by presenting a model built from a corpus that includes morphological and syntactic annotations. The simultaneous presence of unannotated elements and different annotations at the same time in the model makes it possible to constrain nearest neighbour queries to specific types of elements. The model can thus efficiently answer questions such as *What do we eat?, What can we do with a skeleton? What else do we do with what we eat?*, etc. Error analysis reveals that the model can highlight errors introduced into the annotation by the tagger and parser we used to generate the annotations as well as lexical peculiarities in the corpus itself, especially if we do not limit the vocabulary of the model to frequent items.

Keywords: word and annotation embedding, annotation, lexical resources

1. Introduction

An efficient implementation of two simple feed-forward neural network architectures, CBOW (continous bag-of-words) and skip-gram in the word2vec[1] tool (Mikolov et al., 2013a; Mikolov et al., 2013b), has revolutionized the creation of distributional semantic models. The networks are trained to predict a target word from the context (CBOW) or vice versa (skip-gram). The embedding vectors can be extracted from the middle layer of the network and can be used alike in both cases.

The fastText[2] algorithm (Bojanowski et al., 2017) extends the implementation of word2vec by creating vector representations for character n-grams in addition to words. When training this model, the context of a target word (and of its n-gram fragments) is considered to consist of not only other words but also n-gram fragments of those words.

However, all these implementations are trained on a corpus that is a simple sequence of tokens. It is possible to replace words in the token sequence with some sort of annotation, e.g. lemmas, or lemmas followed by morphological annotation (Novák and Novák, 2018), but in that case the representation of word forms will not be part of the model. Although models built from a corpus including grammatical annotation perform better in certain tasks than models built from raw corpora (Ebert et al., 2016; Novák and Novák, 2018), in most NLP tasks, the vector representation of the surface forms of words is also needed. An alternative approach is presented by Levy and Goldberg (2014), where the skipgram model is modified to predict syntactic dependency relations of the focus word and the words attached to it by those relations rather than neighboring word forms.

In this paper, we present a modified version of the fastText implementation of the CBOW algorithm, called CBOW-tag.[3] This implementation can be trained on an annotated corpus and is able to build the vector representation of word forms, lemmas, and any type of additional annotation assigned to lexical items in the training corpus at the same time. This makes it trivial to define the distance between these different types of objects, offering the possibility to answer questions about lexical items having a certain type of relation to each other. For example, we expect a list of food items as a result of a query for nouns related to the verb *eat* through an object relation. In this paper, we present a model built from an annotated English corpus. The method, however, is language independent. Our approach differs from that of Levy and Goldberg (2014) in that a) we use neighbouring words as context, b) include morphologically analyzed/tagged forms in the model as well and c) all representations (raw word forms, morphologically annotated lemmas and syntactic relations) share the same vector space model. This makes it possible for us to formulate queries that constrain part of speech or some morphological features. Morphologically analyzed forms are lemmatized as well. This may seem to make little difference in the case of English, however, it does make quite a difference in the case of morphologically rich languages.

2. Preparation of the corpus

In our experiments, we used the 2.25-billion-token English Wikipedia dump[4]. We applied the neural part-of-speech tagger and dependency parser in the SpaCy framework[5] to the raw text. Each token in the corpus was annotated with lemma, part of speech and a dependency relation to another token. The initial parse is a tree, however, we transformed the initial dependency relations, and added extended ones.

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

[2] https://fasttext.cc/

[3] Our implementation is available at https://github.com/ppke-nlpg/fastText_factored-cbow.

[4] downloaded from https://dumps.wikimedia.org/ in May 2016.

[5] https://spacy.io/

The subject of a passive verb got the same annotation as the object of its active pair, likewise a noun modified by a past participle or an infinitive. Dependency relations were percolated to the heads of coordinate phrases. The head of all NP's attached to verbs in the corpus and their conjuncts was explicitly annotated with its relation to the head verb. The annotation of phrasal verbs and prepositional arguments jointly contains the verb and the particle/preposition. The same applies to nominal predicates with copulas.

In the next step, this representation was turned into a format in which word forms are followed by a series of tags: the first one is the lemma and part-of-speech. In the case of verbal arguments and adjuncts, this is followed by (possibly more than one) tag indicating the predicate and the dependency relation attaching the word to it (see Figure 1).

3. The modified CBOW algorithm

In the annotated corpus used for training the embedding models, word (and punctuation) tokens are followed by an arbitrary number of special tag tokens marked by a ■ symbol as shown in Figure 1. We modified the CBOW algorithm of the fastText framework so that it can use this input to build a model containing the representation of word forms and annotation at the same time.

In the first version, we implemented, we only used word forms as context in the input of the neural network when training the model. As target tokens to be predicted by the model, both surface forms and tags were used as shown in Figure 2.(a). This configuration, however, resulted in a model that did not at all resemble what we expected. The vector representation of tags turned out to be almost orthogonal to the words they were originally related to. They were nearly orthogonal to all other words too. This happened because the negative sampling algorithm encountered these tags only as negative examples in the context of any word. Thus, the network pushed them as far as possible from the representation of real word forms often seen as positive examples. That this was the problem was made clear by an experiment in which we trained a model on a corpus where each word had exactly one tag: the word itself. In this model, the cosine similarity of each word and its tag was about zero.

We eliminated this anomaly by sampling words and tags in the context with a uniform distribution, as shown in Figure 2.(b). Thus, both words and tags were seen as positive and negative training samples and the resulting model now worked as expected.

When training our models, we built 300 dimensional vectors using fastText without the character n-gram option. We used the default parameters, i.e. 5-token window size, frequency threshold set to 5 for both words and tags and negative sampling with 5 samples.

4. What can this model be used for?

The model stores the representation of the surface form, the lemma, the part of speech of words and the typical dependency relations between them in a compact form. The fact that these representations are present in a single model makes it possible to ask questions like *What do people drink?, What do people mine?, What do we believe in?, What can leak?, What do we do with a skeleton?*, etc. However, in lack of a gold standard dataset for the task, evaluation of the model's performance answering such questions is difficult. Thus, instead of performing a comprehensive quantitative evaluation, we had to rely on the result of a human evaluation scenario where we manually evaluated the answers of the model queries about a relatively small set of lexical items.

rank	item	similarity	frequency
0	_drink#VB@dobj	1	18814
1	drink#NOUN	0.7049	36413
2	drinking#NOUN	0.6033	27063
3	juice#NOUN	0.5734	14046
4	beer#NOUN	0.5699	41032
5	bottle#NOUN	0.5634	32186
6	drinker#NOUN	0.5593	3204
7	brandy#NOUN	0.5588	2957
8	champagne#NOUN	0.5379	3691
9	alcohol#NOUN	0.5374	54060
10	pint#NOUN	0.5339	2581

Table 1: Nearest neighbours of the item "objects of the verb *drink*" (_drink#VB@dobj)

item	simil.	freq.	item	simil.	freq.
can	1	2176270	can#NOUN	1	9460
can#VERB	0.9967	2314044	cans	0.9386	5678
may	0.8407	1367560	bottle#NOUN	0.7667	32186
may#VERB	0.8396	1624916	bottles	0.7545	12914
could#VERB	0.8212	963212	bag#NOUN	0.6997	33140
could	0.8167	943772	bags	0.6971	12483
must	0.8164	389083	bottle	0.6969	19000
must#VERB	0.8126	395927	tins	0.6909	783
will	0.7895	1220730	carton#NOUN	0.6866	1506
will#VERB	0.7882	1303870	bag	0.6566	19570
should#VERB	0.7329	594608	containers	0.6511	10963

Table 2: Nearest neighbours of the surface word form *can* and the noun *can*

In order to perform this evaluation, we used a web interface to query the model.The interface displays nearest neighbours (NN) of the query item with their corpus frequency and cosine distance. It is possible to define filters using regular expressions to apply on the NN list. For example, in the case of the question *What do people drink?* we can query nearest neighbours of the item "object of *drink*" (the tag *_drink#VB@dobj* in the annotated corpus) applying a filter returning items with the `NOUN` PoS tag only. The result can be seen in Table 1.

As it was shown in (Novák and Novák, 2018), PoS disambiguation of lemmas also makes a significant difference, as homonymous lemmas of different parts of speech are often not represented well by a single shared representation that is totally dominated by a most frequent sense (see Table 2). E.g. since the auxilliary sense of *can* is more than 500 times more frequent than its noun sense , the representation of the surface word form *can* shows no trace of the noun sense. (The cosine similarity of *can* and *can#NOUN* is 0.27, *can#NOUN* is at rank 5482 on the NN list of *can*.) The representation of *can#NOUN*, on the other hand, very well captures the meaning of the noun having food containers as nearest neighbors.

4.1. Queries on typical arguments, similar verbs and typical actions

Despite the errors introduced by the annotation tools we used, the results returned by the model seem reasonable,

```
The ■the#DET Bryozoa ■bryozoa#PROPN ■_know#VB@dobj ■_be_phylum#VB@nsubj , ■,#PUNCT also ■also#ADV known
■know#VERB as ■as#ADP the ■the#DET Polyzoa ■polyzoa#PROPN ■_know#VB@prep_as@pobj , ■,#PUNCT Ectoprocta
■ectoprocta#PROPN or ■or#CCONJ commonly ■commonly#ADV as ■as#ADP moss ■moss#NOUN animals ■animal#NOUN
■_know#VB@prep_as@pobj , ■,#PUNCT are ■be#VERB a ■a#DET phylum ■phylum#NOUN of ■of#ADP aquatic ■aquatic#ADJ
invertebrate ■invertebrate#ADJ animals ■animal#NOUN . ■.#PUNCT
```

Figure 1: The annotation of a sentence after the second preprocessing step

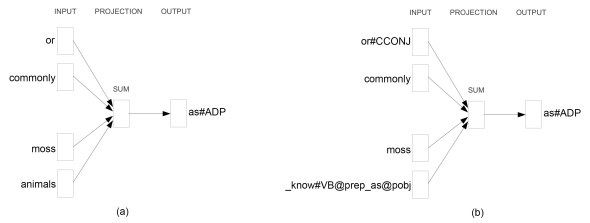

Figure 2: The modified CBOW model architectures

rank	item	similarity	frequency
0	_eat#VB@dobj	1	78427
1	meat#NOUN	0.5810	50211
2	meal#NOUN	0.5749	33003
3	eating#NOUN	0.5535	3159
4	food#NOUN	0.5519	254592
5	manjuu#NOUN	0.5519	5
6	flesh#NOUN	0.5492	15264
7	carrot#NOUN	0.5461	4247
8	gebrocht#NOUN	0.5435	21
9	diet#NOUN	0.5392	35798
10	$\phi\alpha\gamma\epsilon\iota\nu$#NOUN	0.5374	6

Table 3: The nearest noun neighbours of the item "object of *eat*" (_eat#VB@dobj)

especially in cases where the words appearing as a specific argument of a verb belong to a semantically well-defined set. Table 3 lists the nearest noun neighbours of "the object of *eat*". The list is dominated by food names, among them (ranked quite high) the spelling variant *manjuu* of *manjū*, which occurs only 5 times in the corpus, always as the object of eating. It thus gets more tightly associated with eating than more prototypical edible items, about which Wikipedia includes more information like various aspects of preparing, making, serving them. The algorithm also highlights some annotation errors: some foreign words meaning *to eat*, such as the Ancient Greek $\phi\alpha\gamma\epsilon\iota\nu$ or the Finnish *syödä* also appear on the list. These result from "erroneous" annotation of etymologies in Wikipedia (e.g, *Greek "$\phi\alpha\gamma\epsilon\iota\nu$" to eat*, a structure analogous to *some food to eat*. It is not surprising that the underlying parser fails to distinguish the metalinguistic use of the construction. The transformation of some syntactic relations thus seems to have introduced new errors in addition to the ones introduced by SpaCy tools. Filtering low-frequency items from the result lists eliminates the majority of these errors, though.

On the other hand, if we query the model for nouns near the 'subject of *eat*' item, we don't get a very clean list. This is not only caused by the diversity of eaters, but also by the fact that in the corpus with encyclopedic knowledge the subject of *eat* seldom appears as a lexical item.

Certain verbs like *leak*, however, have a more well-defined set of subjects. Clustering its top 100 nearest neighbours shows that it is usually liquids, gases, containers and pipelines, pressurizers etc. and information (secrets, notes) that leak.

Even though for questions like *What do we eat?*, we could query the analyzed corpus directly for words that are linked to *eat* by an object relation and count them, we can also query our model for the answer to more complex questions, such as *What verbs have similar objects or prepositional objects to objects of eat?* as shown in Table 4. Here, the results are filtered to include only (1) direct objects of verbs and (2) prepositional objects of verbs only, respectively. If we compare the result of these queries with nearest verb neighbours of the verb *eat* itself, we can observe that while the latter includes verbs pertaining to physical necessities other than eating (e.g. sleeping), these are replaced by verbs related to destruction, demonstrating that we destroy what we eat. Filtering for prepositional arguments returns brand new items related to eating and consuming that are impossible to collect from a simple embedding model due to the differences in syntactic distribution.

We can query the model for verbs a noun is typically related to, such as what we usually do or what usually happens to a specified object. Table 5 shows the top of the list of verbs that have *skeleton* as their object. We get verbs related to excavation, funeral, reconstruction, but also *fossilize* and *char*, which are intransitive verbs. This error is due to our incorrect assumption that nouns modified by past participles are the object of the original active verb. They can also be patient subjects. And actually the question about what usually happens to a skeleton really amounts to seeking verbs the patient of which can typically be a skeleton.

4.2. Results

Due to the lack of an appropriate gold standard dataset, we evaluated the precision of the result lists returned by the model for a relatively small set of queries. We tested 5 types of queries: (1) nouns appearing as the object of a given verb, (2) verbs having similar objects to that of a given verb, (3) verbs having similar prepositional arguments to objects of a given verb, (4) nouns appearing as the subject of a given verb, (5) verbs having the given noun as their object. We randomly chose 20 relatively frequent verbs[6] and 20 relatively frequent nouns[7] from the model and for each

[6]Verbs checked for the *dobj* relation: *eat, drink, mine, prove, build, excavate, terminate, expect, cause, raise, obtain, end, recognize, organize, read, set up, display, celebrate, address, indicate*; Verbs checked for the *nsubj* relation:*eat, leak, explode, prove, flow, dry, help, change, know, stand, compete, fight, kill, own, found, mention, be high, flee, exceed, sail*

[7]Nouns checked for the *dobj of* relation: *skeleton, rice, toy, key, lamp, paper, lamb, future, stadium, custom, cattle, casualty, lover, rhythm, spelling, civilian, expectation, spectrum, vacancy, uranium*

item	sim	freq
_eat#VB@dobj	1	78427
_eat#VB@dobj	1	78427
_consume#VB@dobj	.6521	34141
_drink#VB@dobj	.6255	18814
_ingest#VB@dobj	.5984	3965
_cook#VB@dobj	.5980	10631
_swallow#VB@dobj	.5414	5390
_eat_out#VB@dobj	.5316	116
_devour#VB@dobj	.5242	3080
_digest#VB@dobj	.4930	2547
_eat_up#VB@dobj	.4925	472

item	sim	freq
eat#VB	1	109537
drink#VB	.6370	39160
consume#VB	.6151	44994
cook#VB	.6138	25875
devour#VB	.6011	4366
chew#VB	.5581	6142
vomit#VB	.5555	3111
ingest#VB	.5551	5736
sleep#VB	.5460	45645
swallow#VB	.5455	10402
munch#VB	.5348	210

item	sim	freq
_eat#VB@dobj	1	78427
_feed#VB@prep_on@pobj	.5829	40039
_feast#VB@prep_on@pobj	.5103	588
_taste#VB@prep_like@pobj	.4741	577
_subsist#VB@prep_on@pobj	.4582	1393
_regurgitate#VB@prep_by@pobj	.4516	9
_dine#VB@prep_on@pobj	.4512	163
_consume#VB@prep_as@pobj	.4512	1050
_cook#VB@prep_like@pobj	.4508	145
_be_rice#VB@prep_along@prep_with@pobj	.4491	11
_forage#VB@prep_for@pobj	.4439	1648

Table 4: (1) verbs having most similar objects to that of *eat*, (2) nearest verb neighbors of *eat*, (3) verbs having the most similar prepositional objects to the object of *eat*

rank	item	similarity	frequency
0	skeleton#NOUN	1	18934
1	_unearth#VB@dobj	.4950	5492
2	_discover#VB@dobj	.4943	156506
3	_excavate#VB@dobj	.4924	12528
4	_find#VB@dobj	.4643	882721
5	_disarticulate#VB@dobj	.4513	12
6	_fossilize#VB@dobj	.4504	68
7	_uncover#VB@dobj	.4426	18932
8	_derive#VB@dobj_that@prep_of@pobj	.4402	12
9	_excavate_up#VB@dobj	.4284	6
10	_mummify#VB@dobj	.4098	144

Table 5: List of verbs that have *skeleton* as their object

query	precision
object>noun	0.85
object>obj of other verb	0.95
object>prep. obj of other verb	0.76
subject>noun	0.71
noun>object of what verb	0.82
all	0.83

Table 6: Precision @40 of NN lists for the 5 tested query types

question and test word we evaluated the first 40 items of the result lists manually. An item in the result list was considered correct if the given word was appropriate in the tested relation (e.g. for the verb *mine*, both *mineral* and *pit* were considered correct for the object relation), or if there is a correct and typical argument that can also appear as a typical argument of the other verb. Words related to the query word with a relation other than what we queried for were deemed incorrect, e.g. water does typically flow in a *canyon* or *valley*, it is not the *canyon* that flows).

The results were checked by 3 human annotators and only those items were counted as correct that were judged to be correct by at least 2 of them. The results are shown in Table 6 for each query type and for an aggregated precision for all queries. As it can be seen, worst results were achieved for the queries regarding the subject relation (see Section 4.1.), while the best results were achieved for one of the most complex query types, i.e. "which other verbs have similar objects".

5. Conclusion

In this paper, we presented an algorithm that generates distributional representations of surface word forms, lemmas and their annotation in a common vector space and a specific model built using the algorithm. In the model presented, the annotations we used were morphosyntactic and dependency relations. Since the representation is compact, the model can be used to answer complex questions like *What else do we usually do with things that we usually eat?* very simply and rapidly. The algorithm, however, can be applied to other types of annotation as well.

Acknowledgments

This research was implemented with support provided by grants FK 125217 and PD 125216 of the National Research, Development and Innovation Office of Hungary financed under the FK 17 and PD 17 funding schemes as well as through the Artificial Intelligence National Excellence Program (grant no.: 2018-1.2.1-NKP-2018-00008).

6. Bibliographical References

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Ebert, S., Müller, T., and Schütze, H. (2016). LAMB: a good shepherd of morphologically rich languages. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, USA, November.

Levy, O. and Goldberg, Y. (2014). Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, Baltimore, Maryland, June. Association for Computational Linguistics.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119.

Novák, A. and Novák, B. (2018). POS, ANA and LEM: Word embeddings built from annotated corpora perform better. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing: 17th International Conference, CICLing 2018*, Hanoi, Vietnam, March. Springer International Publishing, Cham.