Universität Tübingen
- Philosophische Fakultät -

Embedding-based approaches to prediction of semantic compatibility

Submitted to the Department of General and Computational Linguistics In Partial Fulfillment of the Requirements for the Degree of Bachelor of Arts

Author:
Peter Schoener

Advisor:
Dr. Daniël de Kok

October 14, 2018

Eberhard Karls Universität Tübingen

Plagiarism Statement

Contents

1 Introduction

In response to growing interest in applying distributional approaches to problems that require consideration of entire phrases rather than just words, and in order to predict more accurate embeddings than would be found distributionally for compound words, several methods have been developped to combine embeddings into new vectors of the same dimensionality, effectively finding what the embedding would be in the same space for the two words considered as a unit. Parsing is immediately related to composition, since the items being composed would be those which together make up a phrase.

Of course, when training compositional models, the goal is to predict an embedding. This must be done using positive samples, since it is impossible to generate gold standard negative targets other than noise, which would clearly harm the ability to predict valid compositions. Not only does this mean the composition models are unable to meaningfully predict invalid compositions, these would likely not be useful in the first place. The more advanced models are not associative or commutative, so even as part of a larger phrase a prediction for an invalid pair would make the final output less, if at all, valid. Indeed, a meaningless composition is one of the better possible outcomes in this case —the embedding might also collide with an unrelated word or composition, causing misleading results.

It is therefore important to have an accurate parse so that the resulting composition is useful, or to filter pairs that can not be composed after parsing but before composition. By checking one step earlier and looking at possible transitions or terminals when parsing, it would be possible to get a higher parse accuracy. This would be not only helpful for composition as explained above, but also useful in any other domain requiring accurate parsing.

Existing work on compatibility centers mainly on mutual information and related statistical metrics, which are only applicable to known pairs; without at least one co-occurence these metrics generally become undefined. Embeddings, on the other hand, do not even require the individual word to have been seen, or at least not as often. With subword composition methods such as FastText, embeddings can be generated for words not seen at all (Bojanowski, Grave, Joulin, and Mikolov, 2017), and, per the distributional hypothesis on which all embedding methods rest, those seen rarely can be trained if similar words are been seen sufficiently often.

The goal of this project is therefore to create a model that can determine whether a given embedding pair is semantically compatible. Because it is a first attempt at such an approach, the scope is limited, but given the results the models can hopefully be generalized to arbitrary embedding pairs of known types.

Noun-adjective pairs are a natural starting point for a few reasons. They are easy to annotate and extract correctly, giving a clean dataset. They are also generally clearly valid or invalid. Having first explored the problem through this case, we moved to something more directly applicable to a problem that justifies this project: verb-direct object pairs, which would be useful for parsing.

We tested several different models of increasing complexity, the goal being to find the simplest architecture that can achieve high accuracy. First, logistic regression and a feed-forward neural network predicting mutual information, with a cutoff trained to predict validity, which were abandoned in favor of cutting out the restriction of predicting mutual information. That is, again logistic regression and a feed-forward neural network, but without the intermediate layer. Lastly, we adapted two proven composition models, FullLex and transformation selection, to binary classification.

First, however, we investigate existing work on compatibility and the cutting edge of composition, from which the reasoning for our approaches becomes clear. These approaches can then be tested against each other, and tentatively broadened to more varied word and embedding types.

2 Related Work

2.1 Statistical approaches to compatibility

Existing work on compatibility has been primarily on the basis of statistical metrics applied to concrete words. Conditional probability and cooccurrence have been shown to be effective (Volk, 2002), and the lexical association metric (Hindle and Rooth, 1993) also works. Pointwise mutual information can likewise be adapted to a highly effective determiner of compatibility (van Noord, 2007).

The earliest of these works was that by Hindle and Rooth, and deals with prepositional phrase attachment. That is, it looks at ambiguities parallel to that in the famous sentence “I saw the man with the telescope,” where “with” can attach either to the speaker or the man being seen.

They propose an elegant solution that captures a few natural intuitions. The absolute value reflects the confidence of the judgement (on a logarithmic scale). All lexical association (their proposed metric, shown in 1) scores above zero suggest verb attachment, and those below suggest noun attachment.

$$\mathrm{LA}(verb, noun, prep) = \log_2 \left(\frac{f(verb, prep) \cdot f(noun, NULL)}{f(verb) \cdot f(noun, prep)} \right) \tag{1}$$

However, it has a few pitfalls. Firstly, it is only a comparison metric that can disambiguate between two possible attachments, rather than determining absolute compatibility. Secondly, it requires that the noun-preposition attachment have been seen, as have both candidates in isolation, else the metric becomes undefined. Thirdly, it requires that the verb attachment and the unattached noun have been seen or else the metric assigns a value of complete uncertainty. Lastly, even when these have all been seen, performance remains low when they have not been seen very often, as addressed in the paper.

The cooccurrence metric, proposed by Volk, 2002 and shown in 2, has less trouble deciding between the attachment candidates, but is still ultimately a comparison rather than an absolute metric. However, it achieves great accuracy while retaining the speed benefits of a statistical approach. With over 90% decidability and 80% accuracy on those decidable cases, it sets as a precedent an accuracy that can be considered a target to maintain for further work, which could expand to cases beyond the ambiguous ones, i.e. to more general attachments and unseen tokens.

$$\mathrm{cooc}(word, prep) = \frac{f(word, prep)}{f(word)} \tag{2}$$

Another important conclusion, arguably more central to the point of the same paper, is that unsupervised approaches perform equally well as supervised ones for this sort of task. This opens up the possibility of using massive amounts of automatically annotated data rather than a limited manually annotated set. More concretely, this suggests that the type and volume of data needed to effectively train good embeddings and generalizeable transformations for said embeddings is indeed usable for this application.

Similar methods have also been adapted to absolute metrics (van Noord, 2007). The association score, shown in 3, becomes very similar to —indistinguishable from, if the variables are interpreted as the occurrence of each word in its specific role —the PMI of the two words, shown in 4, for which the compatibility is to be determined. While this metric was not tested on its ability to predict whether or not a pair is at all compatible, it does perform very well for disambiguating verb-subject-object attachments.

$$\mathrm{I}(r(w_1, w_2)) = \log\left(\frac{f(r(w_1, w_2))}{f(r(w_1, \_)) \cdot f(\_(\_, w_2))}\right) \tag{3}$$

$$\mathrm{PMI}(x, y) = \log\left(\frac{f(r(x, y))}{f(r(x, \_)) \cdot f(r(\_, y))}\right) \tag{4}$$

Note that this makes it, as tested so far, also a relative metric. However, unlike the others thus far mentioned, it should ideally assign a higher score to any compatible pair than to any incompatible pair, at least for a given verb around which the disambiguation is to be done. This means that at the very least it should be possible to find a threshold above which a noun can be considered compatible with a given verb. Whether this threshold is consistent across verbs is not important to the paper and therefore not investigated, but would be equivalent to the metric being applicable without further modification as a classifier of compatibility or incompatibility.

While not using the exact same measures, there has been research into using similar statistical methods to determine compatibility of a single candidate in a given type of relation, and indeed relatively simple measures of cooccurrence seem to work quite well for this task (Yang, Su, and Tan, 2005). However, the particular domain in question was English pronoun attachment, which is not as broad as the other types of compatibility discussed above. Nonetheless, the good performance of the model means that finding a single cutoff for cooccurrence across the task is not ruled out as a sensible approach.

The exact measure used here is not as directly related to mutual information as those used by, for example, van Noord, incorporating the rank of a candidate within the space of cooccurrences between the given and all competing candidates. Note that this is still not the same thing as training separate models for each element of one of the categories of words: the rank is used merely as a factor, weighting a measure similar to pointwise mutual information. Thus, while it draws on a slightly more complex measure, this may suggest that pointwise mutual information or a similar measure could be used to set a single cutoff.

Given that this is the case, a metric which ranks pairs by co-occurence should be applicable to this task, possibly allowing a simple threshold to be set in order to discriminate between valid and invalid pairs. The question remains, however, which metric should be used. Normalized PMI has a few useful properties for this sort of task (Bouma, 2009), most importantly that it counteracts PMI's bias toward rare collocations and, as a direct result of its nature, has a fixed interpretation; once the cutoff is determined, the space of accepted tokens can be expanded without completely retraining the model. Of course, since this project is designed to work on word embeddings, this would mean first training a predictor for NPMI and then finding the appropriate cutoff.

2.2 Embedding composition

As far as embedding-based approaches, the only existing work is in composition itself. Two notably successful architectures for compostion are FullLex (Socher, Huval, Manning, and Ng, 2012) and transformation selection (Dima, de Kok, Witte, and Hinrichs, 2018). Both of these involve applying a transformation to each constituent and then another to the concatenation of the outputs, the difference being in whether the transformations are trained to be general or word-specific.

Assuming one can accurately predict what the composition should be if the constituents are compatible, it stands to reason that a valid angle might be checking the resultant embedding for validity. The ideal method for the latter operation is less well-explored. However, since the goal here is to take a neural approach, it may be possible to train adaptations of the aforementioned composition models to simply output a binary classification directly or with one extra layer.

Embeddings can be trained to represent different properties of words. While it is clear that with a large corpus a high dimensionality and cutoff, e.g. the common values of 300 and 50, respectively, will lead to more accurate embeddings, this does not mean the represented information is the information we are looking for. The paper introducing GloVe (Pennington, Socher, and Manning, 2014) suggests that smaller windows will favor syntactic information, whereas larger windows favor semantic information. Although our experiment uses Word2Vec embeddings, we apply this intuition to our model as well.

Whether any types of embeddings work better than others and for which tasks is still an ongoing debate, but there is some evidence to suggest that

Word2Vec captures more semantic information than GloVe or certain other methods (Schnabel, Labutov, Mimno, and Joachims, 2015). While this is admittedly more a rationalization for than a factor in our decision to use Word2Vec, it does at least confirm (beyond what is already implicitly shown by the clear effectiveness of Word2Vec in the composition methods referenced) that this configuration is a valid starting point.

3 Preprocessing

3.1 Corpus

The corpus used in this experiment is made up of articles published from 1986 to 2009 in the German Tageszeitung (taz) newspaper. It contains 28.9 million sentences and a total of 393.7 million tokens, tagged and parsed automatically. This is done with the Citar tagger, the parser presented in de Kok and Hinrichs, 2016, and the Lemming lemmatizer (Müller, Cotterell, Fraser, and Schütze, 2015).

The tagger manages 10-fold cross-validated accuracy of 97.41% on the TüBa-D/Z corpus, which is a subset of the taz corpus being used here. The parser achieves just over 90% LAS, and the lemmatizer is about 98% accurate. It is important that the annotations be not only valid but complete so as to ensure that (1) the embeddings on which the model trains are indeed those of adjectives and nouns, (2) as many positive edge cases as possible are accurately modelled (a) so as to make the most meaningful impact on parsing and (b) so they are not caught by the negative sampling, actively degrading the accuracy of the model.

In terms of the data relevant to our experiment, there are 26.6 million noun-adjective pairs found in the corpus for which an embedding can be trained for at least one constituent with the 50 occurrence minimum. About 24.6 million of these are actually used in training, that is, both embeddings are present. The number of usable positive samples is the limiting factor in the total quantity of data, since negative samples can be generated ad infinitum (see 3.3) but we want to aim for a balanced set. Thus, we end up with 49.2 million training pairs.

The verb-noun experiment is done with direct objects, of which there are fewer than adjectives. There are 1.9 million positive samples, of which 1.7 million are used in training. As before, this gives us a total training set size of twice that, i.e. 3.4 million. This data is even less noisy than the adjective-noun pairs, at least in terms of part of speech. Whether the nouns are actually direct objects is hard to tell without references to the original sentences.

3.2 Embeddings

The inputs to all the tested models are Word2Vec embeddings trained over the entire dataset. A window size of 10 ensures that the embeddings carry more semantic than syntactic information, and a minimum count of 50 ensures that all the embeddings used have been trained well enough to be reasonably accurate representations. We settled on an embedding dimensionality of 300 to ensure that the networks have as fine-grained information to work with as possible.

The ConLL data is searched for dependencies between nouns of any type and adjectives of any type. Any pairs for which an embedding is not found for either component are discarded since they will be of no use for training. Because any pair found in the corpus can be assumed to be valid, the positive samples on which the model is trained are simply all pairs which occur in the corpus on which it can be trained, i.e. all those for which both embeddings could be generated.

The way positive samples are generated means that the minimum count for the embedding model has a further consequence beyond ensuring validity of the embeddings: the compatibility models will not attempt to learn based on rare words which might have strange or simply incompletely represented compatibility properties. Because the list is not deduplicated, the models learn primarily from pairs where the compatiblity is well-established, and focus on getting reliable results for more common and therefore more important pairs.

3.3 Negative sampling

Of course, it does not make sense to train only on positive samples. Negative samples are generated first by random sampling, but then filtered against not only pairs that did occur, but those which are embedding-wise similar. This means that not only found pairs are avoided, but also those which would likely also be possible as per the embeddings.

In order to avoid doing a quadratic number of searches depending on the number of similar embeddings taken into account, we make a quadratic number of entries in a Bloom filter, which is of course much faster while still being memory efficient. The filter is designed to have a false positive rate of at most 5% , though in reality this will be far lower since the element count is estimated directly from the number of positive samples and the filter does not need additional space for duplicate elements; it is substantially larger and sparser than it needs to be. The process is further sped up by keeping the neighbor lists in a hash map once they have been looked up, which costs memory at preprocessing time, but this step need be done only once and can be used with all the architectures examined in this thesis.

Each negative sample is generated and then tested against the Bloom filter, then scrapped and regenerated if it is found. This allows precise control over the number of negative samples. In order to maintain a balanced training set, I fixed this to the number of positive samples.

3.4 Test data

Test data is assembled from a few different sources. Some samples are invented, and the rest are drawn from human judgements on random elements from the positive and negative sets, as well as randomly generated samples. Unlike the negative training data, these random samples are not filtered against anything. By relying on human judgement misclassifications can be avoided, i.e. outliers that should not be in the positive set and valid random pairs that should not be in the negative set. Just as importantly, tokens incorrectly tagged as nouns or adjectives can be filtered out; errors when classifying these should not be held against the model since they are not valid inputs in the scope of this task.

Drawing on suggestions from the generated positive and negative sets simply makes it more likely that a definitively positive or negative pair will be shown to the judge. This also does not equate to testing on training data, since the judge is given the chance to change the tag. This chance is amplified by specifically also drawing on the pairs that were misclassified during validation.

The judgements were ultimately all made by one native speaker, but the pairs are, as mentioned, clearly compatible or incompatible. Whether they make pragmatic sense is not considered, so unlike the training data some pairs are included which would rarely occur in real text but which make sense.

In order to ensure that the set remains balanced, the longer of the two pair lists (compatible or incompatible) is truncated. However, it turned out that they were already fairly close in size.

4 Models

4.1 Logistic Regression

In order to determine whether it even makes sense to spend time training a slower, more complex, and less transparent architecture, we first tried applying logistic regression to this task. Although slightly less simple to train than linear regression, it is monotonous, so many of the difficulties created by combining nonlinearities are avoided. It also conveniently squeezes outputs into the range $(0, 1)$, allowing us to interpret them as probabilities. However, because it is asymptotic, the standard version of the function will lead to weights continuing to be adjusted after they are good enough, eventually making them grow out of control. In order to prevent this, as it could reduce the accuracy, we used an approximation of the logistic sigmoid that rounds after a certain cutoff. A good value for this turned out to be ±8, i.e.

$$\text{cutoff\_sigmoid}(x) = \begin{cases} \dfrac{1}{1+e^{-x}} & \text{if } |x| \leq 8 \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

This cutoff value was an educated initial guess, but turned out to be be quite optimal when compared to 2, 4, and 10. This makes a certain amount of sense since the sigmoid function's derivative at this point is vanishingly close to 0; starting around here any adjustment tried by the optimizer would likely be very drastic.

In order to maintain balanced batches, the entire test set is shuffled before training. This is done for the other models as well. The inputs to the regression and feed-forward models are the concatenations of the constituent embeddings, i.e. rank one tensors with 600 elements (that is, the input layer is a 600×10000 rank two tensor).

All four of the models examined in this paper use logarithmic loss since it works well with estimations of probabilities. It penalizes confident misclassifications heavily and tries to increase confidence of correct predictions. This works nicely in concert with the modified sigmoid; the loss goes to zero at some point, rather than chasing it indefinitely.

However, accuracy did not improve far beyond a guess in our initial tests, and so we abandoned this approach and moved to architectures that allowed more complex interactions.

4.2 Feed-Forward Neural Network

The final version of this model is a standard multilayer perceptron with three hidden layers of 2000, 2000, and 1000 neurons. More neurons per layer did not yield any significant improvement, nor did more layers. When compared to a single hidden layer with 2000 neurons, this expansion gives an improvement in validation accuracy of about two percentage points. While not a huge leap, this is substantial enough to warrant the expansion. Moreover, by expanding until performance plateaus, it is possible to determine the limit of this approach.

The hidden layers use leaky ReLU activation with a slope of 0.2 for values less than 0 simply because it has been shown in previous experiments to work well, and indeed it gave better results than the hyperbolic tangent we had been using initially. The hidden layers have dropout with 80% retention to prevent overfit, though with such a large dataset this is not a huge risk to begin with. Using less dropout does not improve accuracy either, so it certainly does not harm the model.

For the output layer, we used the same cutoff sigmoid function as in the regression, and for the same reasons.

Training is done using an Adam optimizer with a learning rate of 0.0025 in batches of 10000 samples. As it turns out, a learning rate over 0.005 is too high for the model to converge at all, and performance starts to pick up around 0.003.

The training outputs are simply booleans —one for positive and zero for negative samples. An output of over 0.5 is considered a prediction that the pair is valid, and an output of 0.5 or lower the converse. Because an output of 0.5 means that the model is entirely unsure of how to categorize the pair, and because there are a roughly equal number of false positives and false negatives ignoring this case, we decided it would be more expedient in practical application to just guess than to create a third, uncertain category. The goal here is binary classification, and the confidences are mostly made transparent so that the models can be more flexibly integrated into other programs in the future.

4.3 FullLex adaptation

Because of the success of the FullLex (Socher, Huval, Manning, and Ng, 2012) architecture in composition, it was reasonable to expect that transforming the noun embedding according to a matrix corresponding to the specific adjective and vice versa, with a final transformation on the concatenation of the two resulatant tensors, i.e.

$$W_o \times [W_v \times u; W_u \times v] + b \tag{6}$$

would yield even better results than the feed-forward approach. For simplicity and to ensure sufficient data was passed in, the embeddings of each category were clustered and each cluster mapped to a transformation matrix. The memory cost of training a separate matrix for each word in the known vocabulary was also untenable, with the raw data (not including overhead) on the order of a terabyte.

This approach uses the same batch size and learning rate as the feed-forward model, but was also tried with more dropout, as this is an important component of the similar transformation selection model. However, even retention as low as 10% did not help the transformation matrices generalize. We train 100 such matrices for each the nouns and the adjectives, as this has been shown to be more than sufficient differentiation for this type of approach. Using fewer matrices and more data points per matrix does not improve performance.

This architecture takes considerably longer to train, roughly ten times as long on the same hardware as the feed-forward model. Some of this is due to the cost of looking up each input's cluster. The most sane way to do this would involve evaluating the input layer all at once, but due to the way that the values are passed from Rust to Tensorflow, this is not possible. Instead, the lookup is done in a less parallelizeable way. This may actually be a case where a performance gain could be achieved by switching entirely to Python due to the incompleteness of the Tensorflow Rust API and the cost of the only simple way to do this operation.

One could also do the clustering in Rust, but in order to stay within the constraints of this thesis and not break compatibility with the other models, this will be left as a possible future development. However, it might not be the most immediately lucrative improvement, as shown later in this thesis.

4.4 Transformation selection

A similar approach to FullLex, transformation selection, has been proposed in Dima, de Kok, Witte, and Hinrichs, 2018. Rather than applying only the words' transformation matrices to each other, a large number of general matrices in the form of a rank 3 tensor are applied to each constituent and whichever of these happen to interact strongly with the embeddings then also have the largest impact on the output, and a nonlinearity, in our case again the leaky ReLU, is thrown into the mix, that is,

$$y_{pred} = W_o \times leaky\_ReLU([W_U \times u + b_U; W_V \times v + b_U]) + b_o \qquad (7)$$

This has proven even more effective at composition than FullLex. For this reason, an adaptation of transformation selection might be expected to outperform the adaptation of FullLex. If more accurate compositions can be predicted with this method, and still assuming that this implies it can model the arguably simpler interaction of whether they combine in the first place, it should do so.

For this experiment, the hyperparameters follow the optima outlined in the paper introducing the method. The transformation tensors are each $80 \times 300 \times 300$. They are separate for nouns and adjectives, and preserve, at first, the dimensionality of the inputs. Crucially, dropout with only 10% retention is used; in composition, this level of dropout is needed for good results. This was applied first to random elements and then to entire matrices within the transformation tensors, but performance was the same either way. The provided model is still set up to use dropout on entire matrices.

Because of the rank of the transformation output and the need to reduce it to a single number, two outer transformations are applied. This has the added effect of essentially generating an embedding and then applying a small feed-forward network with two layers to the output, i.e. applying a feed-forward network to determine the validity of a generated embedding.

	validation			test		
	acc	fp	fn	acc	fp	fn
logistic regression	62.1%	16.5%	21.5%	49.1%	6.7%	44.1%
feed-forward neural net	91.3	4.0	4.7	52.8	7.0	40.2
FullLex	79.3	9.4	11.3	52.6	8.0	39.3
transformation selection	60.5	22.2	17.3	50.2	4.6	45.2

Table 1: accuracy of noun-adjective models on validation and test data

5 Results

As mentioned earlier, the output layers use the logistic function because of how well it maps to percentages. Of course, the percentages are just confidences and not the final output per se. However, we leave the confidences slightly transparent by analyzing the validation and test loss so as to better understand how closely the data could be modelled.

Because the task is binary classification, it makes sense to use the accuracy as a metric of performance, keeping an eye on the ratio of false positives to false negatives. The specific cases with which it struggles can also provide insight, of course, but with 300-dimensional embeddings and the nature of neural nets the reasons for misclassification will only be conjecture.

Furthermore, because all the training and validation data is automatically annotated, there are some errors that propagate into the classifier. The rate of things that are incorrectly tagged as nouns or adjectives is higher in the misclassified output than in the input, which bodes well for testing on gold standard data. Because one application for this project is as a step in making parse decisions, there is a bit of a feedback loop here. In order to make sure that it will work correctly and has not been trained too much on the errors, the model also needs to be evaluated on a hand-written list of noun-adjective pairs, both valid and invalid.

The most directly relevant and important measure is, of course, accuracy. However, depending on the task at hand, a low rate of false positives or false negatives might be more desirable, so these are provided as well, though one should note that for the sake of this thesis the models were made to err on the side of caution; absolutely uncertain cases are marked as incompatible. Rather than implement a cutoff for confidence to make a decision at all, we simply document the confidence with all the misclassified samples. Accuracy and error types are broken down in Table 1.

It is immediately apparent that the feed-forward neural network is far and away the most accurate approach. It does not clearly maintain this edge when tested on the human-annotated data, but the others deteriorate to a similar level. Reasons for this are explored later, but refer back in particular to 3.4 for the ways in which the training and test data differ. However, although the test accuracy is not much better than a guess, the model is clearly not just guessing. It generates far more false negatives than false positives, but still some true negatives and false positives. As mentioned, the model achieves a great validation accuracy, which is promising despite the results on the test set.

The FullLex approach performs fairly well, though this came as a bit of a surprise; the first few runs of it gave accuracies in the low 60s, but the random initialization seems to have pushed it out of a massive local minimum when the model was retrained from the beginning. A higher learning rate had not helped previously; the adjustments were too large at that point for the model to remain stable within the optimal range. Regardless, this model clearly outperforms logistic regression, as expected due to its basis in previous work. Unfortunately, it suffers just like the feed-forward network when applied to the test data.

Also surprisingly, the transformation selection approach markedly underperforms the FullLex one; they are fairly similar models, applied here with a similar rationale. Its performance in composition was shown to be on par with and in some cases better than FullLex, so we expected that to be reflected here. The transformation selection model might also have just been unluckily initialized, but as far as we can tell it does not perform better with a different learning rate.

One interesting difference between the transformation selection and the other approaches is that it has a substantially higher rate of false positives than false negatives, despite ties going to the negatives as with the other models. This is hard to explain, but may be an effect of the final layers' function: because it is designed to implicitly train an embedding generator and then evaluate the output for validity, it could be that the final layers, simply due to the nature of how embeddings really are distributed within the space, tend to believe they are valid. If one considers the way that embeddings are trained and the relationships that should exist between dissimilar words, it makes sense that an embedding model will make relatively full use of the space available. This means that there will likely be few pockets that are

	val. loss	test loss
logistic regression	0.6352	14.314
feed-forward neural net	0.2266	36.864
FullLex	0.4600	11.910
transformation selection	0.6416	13.848

Table 2: truncated (for inputs outside of $[-8, 8]$) logarithmic loss of noun-adjective models

semantically invalid areas, and this could extend to generated embeddings as well. Thus, the best guess would often be in favor of the embedding being valid, but in our case half of them are not.

One should also consider how severe the misclassifications are, and, being used as the loss function during training for exactly this reason, the [truncated] logarithmic loss is a good indicator of this. Of course, it still counts correct predictions which are insufficiently confident, but the nature of this function is that it penalizes severely wrong predictions far more heavily. The average loss for each approach is shown in Table 2.

This is especially important in the case of validation loss; given the high accuracy on valiadtion data, the loss may shed some light on which classifiers would be more reliably accurate on a broader test set, perhaps a better engineered one; one would imagine that a model getting high scores with great confidence would generalize better than one doing the same with low confidence.

The results here are in line with what we saw in the accuracy rankings, with the values fairly clearly linked to the number of misclassified samples. This does not mean, however, that this data gives no additional information: It shows that none of the four models misclassifies validation data with vastly more confidence than the others, or at least not on average.

However, the test data shows something else. The test loss is much lower on the other methods than the feed-forward network, despite the accuracy also being lower. This means that while they make bad judgements on the test data, they make them with less confidence. One could be tempted to interpret this as meaningful for applications of this method that do not apply a decision below a certain confidence level, but it is worth noting that the confidence of the misclassifications is still quite high; there might not be very many decisions left to act on with a sufficiently high cutoff.

Given the promising results, it made sense to expand the approaches to

	acc	fp	fn
logistic regression	72.5%	11.3%	18.1%
feed-forward neural net	86.8	3.5	9.7
FullLex	79.1	8.7	12.2
transformation selection	70.5	8.7	20.1

Table 3: accuracy of verb-noun models on validation data

other types of embedding pairs. Since this is meant as a test of generalizability of the proposed models, we did not tune the hyperparameters separately for the new task. This being a secondary task, no human-annotated test set was prepared; the values shown are for validation data. The accuracy of the same architectures, retrained, is shown in Table 3.

The logistic regression worked much better on this set than on the noun-adjective pairs. This combination also yielded the only model to consistently and clearly improve with each of the first several epochs. The transformation selection worked substantially better than on the adjective-noun task as well, and the feed-forward network slightly worse. However, the ranking of the approaches remained the same.

The transformation selection approach no longer had a higher rate of false positives than false negatives, meaning that even if the theory about the distribution of valid values of adjective-noun compositions is valid, it does not also hold for verbs and direct objects. It is, of course, also possible that the other result came about some other way; interpreting the weights in these large and multilayered networks is hardly feasible.

The drastic increase in the performance of the logistic regression and transformation selection approaches is difficult to explain. While this test is looking at only one of roughly three possible kinds of verb-noun attachment and the other at any adjective-noun attachment, this is because there are more ways to attach a noun to a verb than to an adjective; this task is not really more specific or smaller in domain.

That being said, there are far fewer data points on which to train it. This makes sense; a sentence can have any number of adjectives but usually just one direct object, depending on how dependencies between conjuncts are handled and what is considered a sentence. As a result of the smaller dataset, it could be that the negative sampling is less “encumbered” by rare pairings and is able to generate more moderate negative samples. The rate of false negatives relative to false positives increased across the board, possibly because the

	loss
logistic regression	0.5611
feed-forward neural net	0.2898
FullLex	0.4424
transformation selection	0.5685

Table 4: truncated logarithmic loss of verb-noun models on validation data

negative training samples are now less outlandish and the positives are more easily mistaken for them.

If we assume that the negative samples are less extraordinary, it could be that the border drawn by logistic regression is simply not pulled as far out of alignment by these outliers. This would suggest that they might indeed be closer to being linearly separable than in the other experiment, or even that the problem is closer to being linearly separable than indicated by the other test. However, this leads to a somewhat counterintuitive conclusion: it would mean that the negative samples generated by the larger dataset are closer to the boundary plane than those created under fewer restrictions; of those correctly classified, only the x for which $|wx| \leq 8$ cause the weights to change.

As before, the log loss can be a source of insight into the model; it is provided in Table 4.

The loss values here are in line with the rest of the results so far; the overall ranking in terms of loss is the same as in the last three evaluations and the values themselves are close to what they were for the noun-adjective test. This means that the confidence overall of each of the models stayed roughly the same when transferred to the verb-noun task. The most substantial changes in confidence are those of the logistic regression and transformation selection approaches, which is in line with the idea that these negative samples are farther from the decision boundary; they are more easily and more confidently classified as such.

Another potentially interesting test was applying the verb-noun model to the noun-adjective task. While this would clearly not be as accurate as using the noun-adjective-specific model, it sheds some light on the feasibility of training a single more general model. Of course, creating new models from a concatenation of the two training sets would yield a more accurate indication of how effective a general approach can be, but this test can give indications of whether there are deeper patterns that can carry over from one pair type

	acc	fp	fn
logistic regression	48.9%	19.6%	31.5%
feed-forward neural net	52.2	4.6	43.3
FullLex	48.5	21.5	30.0
transformation selection	45.4	25.7	28.9

Table 5: accuracy of verb-noun models on noun-adjective test data

	loss
logistic regression	14.155
feed-forward neural net	12.831
FullLex	15.031
transformation selection	13.765

Table 6: truncated logarithmic loss of verb-noun models on noun-adjective test data

to another, as opposed to training entirely for some sort of compromise.

This test is done on the human-annotated noun-adjective data because it is entirely valid, but also on the noun-adjective training data since it is more in line with what the model was trained on. These results are shown in Tables 5 and 6.

The accuracy is not substantially different than in the other test, but the one indicator that the model was not just guessing, the imbalanced false positives and false negatives, is gone here, the only exception being with the feed-forward neural network. This means that the models clearly do not generalize to verb-noun pairs. Even the feed-forward neural network, which still seemed to not be guessing, can not be confirmed to have generalized.

6 Evaluation

6.1 Data

The training data used in this thesis was quite accurately tagged and therefore the results are for the most part transferrable to practical applications, but it is not perfect. As mentioned before, as parsing improves, potentially even through application of compatibility checking techniques such as those presented here, so will the applicability of these models when trained on such data. While this thesis does not investigate the bounding effect of corpus quality on accuracy, this is an important factor to look at when trying to improve the model; it could be that the model is only being held back by the corpus, but it could equally be that the little noise present is already within the confines of what it can work around entirely.

Even with a perfectly annotated text corpus, the negative sample extraction is not perfect. Firstly, it relies on having a large anount of data, which essentially rules out human annotation, and secondly it allows, even with a large corpus, valid pairs to be added to the invalid set. Not all embedding-wise similar tokens will share the same compatibility characteristics as a given token, meaning some invalid pairs are left indeterminate as far as the training data is concerned. Conversely, not all those which do share the same characteristics will rank among the ten lowest cosines.

One reaction here might be to simply move to a cosine-cutoff-based approach, but synonym extraction has proven far more complex than this in practice. While this is not, strictly speaking, a synonym extraction problem —antonyms, for example, will share the same compatibility properties —it is somewhat similar in that other parts of speech should be ruled out, dense areas need to be contended with, connotations play a role in context, etc. Clearly, a better negative sampling system would require further thought, but it would likely bring improvements, at the very least making the performance in the real world easier to guarantee.

The human-annotated data is in part generated by human judgements of randomly generated pairs. This means that the distribution of even the valid pairs is not in line with that which would be found in real-life text. While the specific incidences of misclassification can be taken as indicators of how well the model works, the accuracy rate does not necessarily reflect the accuracy in real world use. A common but misclassified pair is much worse than a rare misclassified one, and this is not reflected by the test set. This was part of the motivation behind using human judgements of the found pairs and those generated by negative sampling, but this is also not without its issues. While not the same as testing on training data —the pairs may switch categories —it may carry the same issues to an extent, since it does not help test whether the model has learned to generalize to pairs never encountered. This is arguably one of the more important aspects of the task, since certifying that a pair has been seen can be done with much simpler methods.

However, the human-annotated data clearly does not align with the training set. It yields far too many false negatives. This may be a symptom of the negative sampling being too aggressive; it seems that some of the randomly generated pairs would have worked after all. A corpus of this size may still be too small to be able to expect all or almost all valid combinations to be present.

This may seem to contradict the conclusions from the negative sampling on the verb-noun task, but it does not. The dataset can be too small to contain enough positive samples while being large enough —under the methodology used here —to rule out too many negatives.

Moreover, there are some combinations which are semantically valid but largely disqualified by world knowledge or commonly held bias, for example "erschreckend" and "Schwimmweste". A life-saving device will usually not be described as frightening, but in a specific context or when the vest has a particular appearance it might make sense. The extenuating circumstances required to make this match plausible do not make it invalid in general; the hangup here is the bias of almost any reader or writer, not whether that type of adjective can be applied to that kind of noun. This is the case for a great many of the misclassified pairs.

The patterns in the false positives are less clear, though it could be that they arise in edge cases where one word is almost universally compatible, but not in the specific case tested. For example, "tierisch" can be applied to a great many things, though usually as an adverb (it is worth noting here that several verbs slipped into the training set, and that German does not clearly distinguish between adjectives and adverbs), but it can not be meaningfully applied to "Pommes," a pair that was erroneously categorized as valid. Regardless, this is, as already shown, not the type of error to be worried about in the test step.

6.2 Models

One of the models presented in this thesis, namely the feed-forward neural network, performed quite well, but there are still a few things to consider when applying these results in practice, and room for improvement in this and the other methods. As mentioned above, it is difficult to determine whether this will transfer to other datasets with different but still real pair distributions; the test data offers a flatter but ultimately similar distribution. Also, training neural models takes a very long time, especially when separate models are trained for each type of pair. On a modern server with 288 GB of

RAM and a Broadwell Xeon (E52698) using one core of an nVidia Tesla M60, training the simple feed-forward network to peak performance took about a day. For this reason, and because it would make it more transparent what properties of embeddings are indicators of compatibility, improvements in regression models would be worth pursuing.

The composition-based approaches are also worth investigating further; they have been shown to work well on composition and yet performed somewhat worse on this related task. This could be a matter of vastly (since small tweaks gave no improvement) incorrect hyperparameter tuning, or of needing more layers after the composition-based output. The original attempt to model PMI with a feed-forward network and set a cutoff on it with a final layer was abandoned in order to free up the intermediate layers, but maybe with the composition-based approaches this sort of training could be made to work, i.e. essentially attaching a neural validator to the end of an existing composition model.

However, the differing results between the FullLex and transformation selection approaches could suggest that validating compositions is not the most effective way to predict compatibility. Due to the way that the output layer of the transformation selection approach works, it is effectively a feed-forward network validating a composition, but with more freedom to find an optimum since the embedding composition itself is not directly trained. Then again, if might be that, in contrast to the approaches that were originally simplified from this sort of approach, the backpropagation is too complex without the constraint.

When using only one hidden layer, the feed-forward network still performed fairly well, achieving over 80% validation accuracy on the adjective-noun task. Given that this is the case, one should expect the transformation selection to work at least as well; a transformation tensor made entirely of unit matrices would essentially be the same. This means that the backpropagation was not finding the global optimum, so some hyperparameters would need to be nudged in order to find the actual peak performance of this approach.

One should also bear in mind that the composition-based models took substantially longer than even the feed-forward network, with the same number of epochs on the same hardware taking about four times as long for the FullLex-based architecture, and the transformation selection slightly longer than that. However, they did not benefit as strongly from multiple passes over the data, meaning that one or two epochs suffice in order to reach peak accuracy. This roughly evens out, but that may not be true on smaller datasets since the composition-based models have vastly more parameters to train.

6.3 Results

It is worth noting that the original experiment was designed to test compatibility of words that can only be attached to each other in one way. The second experiment, meanwhile, focusses on whether a noun can be attached to a verb as a direct object, which is particularly useful for disambiguation in parsing (see again van Noord, 2007), but is a somewhat different task since it enforces a specific kind of compatibility.

The results on the two tasks, when compared, give a bit of insight into the effects of the negative sampling approach used here. While this is not the focus of this thesis and could likely be investigated to the point of becoming a thesis of its own, it is an important part of embedding compatibility checking, and so worth having a quick look at. It seems that the negative sampling approach resulted in saner negative samples when done on a smaller set; there is very little noise in terms of incorrectly tagged words in the verb-noun negative set. The adjective-noun set, on the other hand, seems to have had this noise amplified by excluding so much of the embedding space.

One should note here that the amplified noise is definitely not something that would have been created by completely random sampling; since the word lists were not deduplicated, these misclassified words can only be selected with a rather low probability. They are still not the majority of the output of course, but they are substantially more common in the noun-adjective set than in the verb-noun set. Of the former, they are again more common as negative samples than as positive ones, which again points to this as a likely cause.

As for the validation accuracy, recall the results of Volk, 2002 mentioned earlier. While that project focussed on different types of pairs than this thesis, concerning itself with prepositional phrase attachment, and while it was a comparison-based approach, it set a precedent of 80% accuracy on the most decidable 90% of samples. The task may indeed be more difficult when both attachments could be valid (or when the model predicts that neither should), but the results shown in this paper are promising in terms of their potential to help improve that score.

7 Conclusion

The goal of this thesis was to put forth a proof-of-concept model that used a neural approach to determine semantic compatibility. Indeed, it is possible to predict, with a high degree of accuracy, the compatibility of nouns with adjectives. However, this could not yet be transferred to a more varied test set. The negative sampling is clearly not aggressive enough, but simply amplifying the current method would be too indiscriminate. This means that a more precise method for negative sampling is needed, or a large dataset including human judgements of a large number of incompatible pairs.

However, given the validation accuracy, these methods, especially the multilayer perceptron, seemed very successful at first. It is likely that, given more consistent training and test data, they would perform quite well.

Another important conclusion to draw from this work is that this is a substantially different task from embedding composition itself. If methods based in composition are to be adapted for this purpose, they will need to be more extensively modified rather than just ported. As it stands, they could not be trained to as high an accuracy as simpler methods, even on a corpus that has been shown (Dima, de Kok, Witte, and Hinrichs, 2018) to be plenty for training compositional models.

We also showed that these methods transfer well to other types of pairs, though they need to be retrained. Whether a general compatibility checker can be trained remains to be tested in future work. However, it would be reasonable to assume that these methods could be extended to composed embeddings. The feed-forward approach may be generalizable to multiple types of pairs with one set of parameters, but this would require further investigation and very likely a larger network.

With regards to the negative sampling, it seems, as mentioned above, that too much of the space was excluded by the larger dataset; one possible future improvement could be to adjust the number of neighbors excluded depending on the size of the dataset, which seems like an easy solution under the standard assumption of a Zipfian distribution of word frequencies.

7.1 Future Work

In terms of actual application, the effect of integrating this filter as part of a parser or compositional model has not yet been tested, so we can not yet say whether it is or can be effective enough to improve them. This is left as a topic for later research, but this is research that might already be worth pursuing even without an improved negative sampling system or training set. It could be that by advocating against rare parses the models proposed in this thesis would already create a useful feedback loop.

Moving one step further, if composition functions are to be applied multiple times, the compatibility checking will need to work either with the entire phrases, which is untenable for even slightly long sequences, let alone long-distance dependencies, or work on the output of the previous compositions. While compatibility of composed embeddings is beyond the scope of this project, it would be a natural extension.

All in all, compatibility seems to be a substantially simpler task than composition, with a highly accurate (though of course the accuracy of composition can not be measured in a comparable way) model achievable using a simple feed-forward neural network. This should only improve with access to better data and the development of more valid negative sampling methods, though the method used here seems to be a decent starting point.

References

[1] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. “Enriching Word Vectors with Subword Information”. In: *Transactions of the Association for Computational Linguistics* 5 (2017), pp. 135–146. ISSN: 2307-387X.

[2] Gerlof Bouma. “Normalized (pointwise) mutual information in collocation extraction”. In: *Proceedings of GSCL* (2009), pp. 31–40.

[3] Daniël de Kok and Erhard Hinrichs. “Transition-based dependency parsing with topological fields”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers).* Vol. 2. 2016, pp. 1–7.

[4] Corina Dima, Daniël de Kok, Neele Witte, and Erhard Hinrichs. “No word is an island — a transformation weighting model for semantic composition”. In: (2018).

[5] Donald Hindle and Mats Rooth. “Structural Ambiguity and Lexical Relations”. In: *Comput. Linguist.* 19.1 (Mar. 1993), pp. 103–120. ISSN: 0891-2017. URL: `http://dl.acm.org/citation.cfm?id=972450.972456`.

[6] Thomas Müller, Ryan Cotterell, Alexander Fraser, and Hinrich Schütze. “Joint lemmatization and morphological tagging with lemming”. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing.* 2015, pp. 2268–2274.

[7] Jeffrey Pennington, Richard Socher, and Christopher Manning. “Glove: Global vectors for word representation”. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP).* 2014, pp. 1532–1543.

[8] Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. “Evaluation methods for unsupervised word embeddings”. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing.* 2015, pp. 298–307.

[9] Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. “Semantic compositionality through recursive matrix-vector spaces”. In: *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning.* Association for Computational Linguistics. 2012, pp. 1201–1211.

[10] Gertjan van Noord. “Using Self-trained Bilexical Preferences to Improve Disambiguation Accuracy”. In: *Proceedings of the 10th International Conference on Parsing Technologies.* IWPT ’07. Prague, Czech REpublic: Association for Computational Linguistics, 2007, pp. 1–10. ISBN: 978-1-932432-90-9. URL: `http://dl.acm.org/citation.cfm?id=1621410.1621411`.

[11] Martin Volk. “Combining Unsupervised and Supervised Methods for PP Attachment Disambiguation”. In: *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1.* COLING ’02. Taipei, Taiwan: Association for Computational Linguistics, 2002, pp. 1–7. DOI: `10.3115/1072228.1072232`. URL: `https://doi.org/10.3115/1072228.1072232`.

[12] Xiaofeng Yang, Jian Su, and Chew Lim Tan. “Improving pronoun resolution using statistics-based semantic compatibility information”. In: *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics.* Association for Computational Linguistics. 2005, pp. 165–172.

Appendices

A Data

For the data used in this experiment, contact Dr. Daniël de Kok.

B Code

As of the time of submission, the code for this project is available at `https://github.com/peterr-s/n-adj-compat`.