UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Maksym Del

Towards Phrase-based Unsupervised Machine Translation: Phrase Representations

Master's Thesis (30 ECTS)

Supervisor: Mark Fishel, PhD

Tartu 2018

Towards Phrase-based Unsupervised Machine Translation: Phrase Representations

Abstract:
Current unsupervised machine translation models despite achieving promising results work quite modestly comparing to the supervised counterparts. This work aims to make a step towards a new research direction of Phrase-based Unsupervised Machine Translation. Since current word-based models rely on representation of words, phrase-based models require appropriate phrase representations. These representations should be learned without supervision, address phrase specific multiword expressions issues, and their embedding space has to follow certain regulations for unsupervised translation to perform reasonable. We specify what makes phrase representations effective in terms of Unsupervised Machine Translation, define and evaluate Unsupervised Compositional Modeling Framework for Phrases, and show how to use this framework to satisfy to the proposed requirements thus obtaining effective representations for phrases. We make the code and trained models publicly available as an open source project.

Keywords:
Phrase embeddings, unsupervised learning, embedding space linearity, multi-word expressions, recurrent neural networks, phrase compositionality, skip-gram, word2vec

CERCS: *P176, Artificial intelligence*

Fraasipõhine juhendamata masintõlge: fraasiesitused

Lühikokkuvõte:
Olemasolevad juhendamata masintõlke lähenemised saavutavad küll lootusrikkaid tulemusi, mis on aga halvemad kui juhendatud masintõlke meetodite puhul. Käesolev töö arendab uut fraaside tasemel töötavat lähenemist juhendamata masintõlkele: kuna praegused sõna-põhised lähenemised masintõlkele kasutavad sõnade vektoresitusi, siis selle uue lähenemise juures on vaja vastavaid vektoresitusi fraasidele. Neid esitusi on vaja õppida juhendamatul viisil, arvestades ka fraasidele spetsiifilisi eripärasid nagu mitmesõnalisi väljendeid, ning lisaks peab esituste vektorruum rahuldama teatud nõudeid, et juhendamata masintõlke töötaks. Antud töö defineerib fraasiesituste effektiivsust juhendamata masintõlke kontekstis, loob juhendamata kompositsionaalse modelleerimise raamistiku fraasidele, ning näitab kuidas raamistiku kasutades jõuda effektiivsete fraasiesitusteni. Arendatud skriptid ja treenitud mudelid on jagatud avatud lähtekoodi projektina.

Võtmesõnad:
Fraasivektorid, juhendamata masinõpe, vektorruumi lineaarsus, mitmesõnalised väljendid, rekurrentsed tehisnärvivõrgud, fraaside kompositsionaalsus, skip-gram, word2vec

CERCS: *P176, tehisintellekt*

Contents

1 Introduction

Most successful approaches to machine translation (e.g. [WSC+16, BCB14, VBB+18, GAG+17]) rely on availability of the parallel corpus. Parallel corpus is a set of sentence pairs, where the first sentence is in the source language, and the second sentence is the translation of the first one in the target language. Supervised Neural Machine Translation employs encoder-decoder architecture, where encoder reads the source sentence and produces its representation which is then fed to the decoder that tries to generate the target sentence word by word. Cross-entropy loss is usually used as a training objective and beam search algorithm is used for inference. These neural models show state-of-the art performance but require large amount of sentence pairs.

On the other hand, there is a Statistical Machine Translation parading that is based on phrase tables that are learned from parallel corpus. These methods are currently replaced by neural counterparts for high-resource languages, but perform better in low-resource settings. ([BBCF16]).

For some language pairs the size of parallel corpus ranges from extremely low to almost zero. These extremely low-resource language pairs were a motivation for the Unsupervised Machine Translation ([LDR17, ALAC17]) that aims to translate language without usage of the parallel corpora for training.

The first step of the unsupervised approach to translation is the same for all methods: learning word level embedding spaces for source and target languages and then aligning these spaces. That is, we first learn vector spaces, where each word of the language has its vector of real values. Next, one of the unsupervised vector space mapping methods ([ALA17, CLR+17]) is applied to align spaces together and perform word by word translation. This mapping is only possible because of the certain properties of the word embedding method that is used to get word vector representations. Lastly, some refining using neural network models is done to improve system's performance. However, current Unsupervised Machine Translation systems, despite achieving promising results, are not on pair with their supervised counterparts.

Phrase-based Unsupervised Machine Translation is a novel task and by the time of writing there is no published work that would propose the solution to it. However, the author of the thesis takes part in research activities pointed towards this direction. Discussing possible end-to-end approaches to the Phrase-based Unsupervised Machine Translation is out of the scope of this work; however, we believe that suitable phrase representations are necessary part in future Phrase-based Unsupervised Machine Translation systems.

Note that Statistical Machine Translation approaches that are based on the phrase representations require smaller amounts of data to achieve reasonable results. However, current unsupervised models does not employ phrase information. Phrase-based Unsupervised Machine Translation is motivated by the assumption that methods working without parallel data can benefit from the usage of phrases as the basic units. In order

for phrases to be used for Unsupervised Translation they have to be represented in a suitable way. Thus, phrase-based view on unsupervised translation introduces specific set of requirements on phrase representations.

First of all, since word-based unsupervised machine translation relies on word embedding, phrase-based machine translation by analogy will require a suitable way to embed phrases. Indeed, word embedding is the well-established method for converting words into fixed size vectors of real numbers that are then used as an input to the neural and other algorithms. There are developed methods for this such as word2vec ([MCCD13]). With the recent advances such as fasttext ([BGJM17]) these method allow to get semantically rich embedding for almost any word. These methods for words are successfully used in the current Unsupervised Machine Translation Systems, but analogues phrase embeddings, on the other hand, are harder to obtain.

Some methods for learning phrase vectors require supervised data (e.g. [CKS$^+$17] to obtain embeddings which is controversial to the idea of unsupervised learning for low-resource language pairs. Other related work ([KZS$^+$15]) relies on corpus of consecutive sentences which is not necessarily available in suitable amounts for some languages. Certain neural approaches ([Bal12]) while avoiding downsides of the other methods, do not satisfy to linearity requirement, which is specific to our task of interest, as we will see next.

Indeed, since unsupervised bilingual word embedding mapping ([ALA17, CLR$^+$17]) is the first necessary step used in all of the existing methods of unsupervised machine translation, phrase-based machine translation approach will require unsupervised bilingual phrase embedding mapping. This implies the linearity constraint ([ALA18]) on phrase embedding vector spaces for both languages. This means if we use an embedding method that learns embedding spaces with non-linear relationships between vectors, we will not be able to apply unsupervised methods to map embedding spaces between languages. In conclusion, our embedding spaces has to be linear.

Lastly, working with phrases introduces issue of multiword expressions. Some word combinations result in meaning that can not be derived from individual words meanings. Examples include named entities like "star wars", "hey jude", or "james bond" and other phrases such as "couch potato". In these examples, individual words (e.g. "couch" and "potato") have nothing to do with the result phrase meaning ("couch potato" is the name for the person who does not work-out and mostly spends her time relaxing at home). In Phrase-based Machine Translation we might sometimes want to translate a phrase word by word, but sometimes we want translate a phrase as a whole. On the one hand, like in case of "couch potato", multiword expressions should be translated as a whole. On the other hand, some expressions like named entities, should not be translated at all, but rather left as they are. Since unsupervised translation relies on bilingual mapping, "Mississippi river" (river in the USA) can for instance be translated as "Võhandu river" (river in Estonia) and "Kersti Kaljulaid" (the president of Estonia) can become "Donald

Trump" (the president of the USA), which we do not want to happen. In both cases we want to detect these multiword expressions; therefore, we need phrase representations to be suitable for the non-compositionality detection.

One approach ([MSC+13]) that could satisfy to all of the above requirements treats phrases as separate single-token vocabulary entries (think of "good_thesis" where two words concatenated with the underscore to form the single-token phrase). We then can apply the same word2vec methods ([MCCD13]) that are used for words. This, however, is harder to do in practice, since we want vectors for all possible phrases. The size of the vocabulary grows almost exponentially over the length of the phrase. Vocabularies of that order of magnitude do not fit into the computer memory in most cases. Moreover, there is a data sparsity problem which means that some phrases may rarely occur even in a big training corpus. This makes hard for word2vec to learn reasonable vectors for such phrases. At the same time, authors of [MSC+13] made an interesting observation that simple addition of word vectors (learned with word2vec) together results in meaningful phrase vectors. The idea to combine word embeddings to get a phrase embedding is a good fit for our needs, but addition operation while effective, may not be optimal way to do this combination. This leads us to the idea of applying machine learning techniques to learn the optimal parametrized function for combining words. We first learn vectors for phrases as single-token vocabulary entries ("good_thesis") as well as for individual words ("good" and "thesis" separately) and then train regression models to predict phrase vectors from their word vectors. This approach while was used as an intermediate step in solving other tasks ([MSY+14, YFH15]), was not considered neither in context of our task nor as a standalone method. We thus study it in the task-independent fashion (see Section 4) and show how the framework satisfies to all constrains that Phrase-based Unsupervised Machine translation might imply on phrase representations.

In summary, this work makes following main contributions:

- We define 3 main requirements that the task of Phrase-based Unsupervised Machine Translations implies on phrase representations.
- We study and present the Unsupervised Framework for Compositional Phrase Modeling.
- We show, both analytically and experimentally, how this framework satisfies to the defined requirements and thus allows to obtain suitable phrase representations.

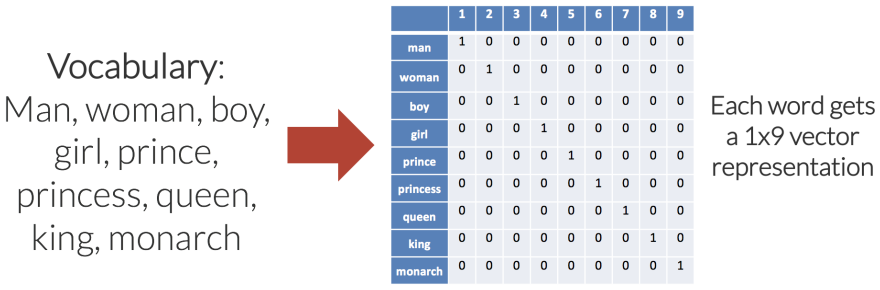
This thesis is organized as follows. Section 2 provides background information necessary to understand requirements that the task of Phrase-based Unsupervised Machine Translations implies on phrase representations. Section 3 then defines these requirements. In Section 4 we present and evaluate the Unsupervised Framework for Compositional Phrase Modeling. Finally, in Section 5 we both analytically and experimentally show that this framework effectively satisfies to the defined requirements..

2 Background

This section gives a background on phrase non-compositionality issue and provides overview of methods that make Word-based Unsupervised Machine Translation possible. We believe this methods will play an important role in the future Phrase-based Unsupervised Machine Translation systems as well.

2.1 Distributed Word Representations and Skip-gram Model

In order to work with the natural language data, one should be able to convert text into the format that is suitable to be the input for the natural language processing algorithms. In most cases we want to convert tokens into the vectors of real values. This procedure is called word embedding. Consider the vocabulary of words and matrix with number of rows equals to the number of words in the vocabulary. Then you just select corresponding row in order to retrieve a word vector representation. Refer to the Figure 1 for example. The only question is how to obtain the matrix where word vectors will correspond to the points in some semantically rich embedding space.



	1	2	3	4	5	6	7	8	9
man	1	0	0	0	0	0	0	0	0
woman	0	1	0	0	0	0	0	0	0
boy	0	0	1	0	0	0	0	0	0
girl	0	0	0	1	0	0	0	0	0
prince	0	0	0	0	1	0	0	0	0
princess	0	0	0	0	0	1	0	0	0
queen	0	0	0	0	0	0	1	0	0
king	0	0	0	0	0	0	0	1	0
monarch	0	0	0	0	0	0	0	0	1

Figure 1. Toy example of the word embedding matrix. Each row of the matrix corresponds to specific word (Figure comes from [Lyn]).

Distributed Word Representations. Distributional hypothesis proposed in [SH54] states that words occurring in similar contexts tend to have similar meanings. This hypothesis became an underlying assumption for the vast majority of state-of-the art word embedding methods ([MCCD13, BGJM17]). These methods result in some useful properties. For instance, word vectors for semantically similar words are grouped together in the embedding space, and dissimilar words have long distances between their vectors. Moreover, algebraic operations on such word vectors correspond to semantic operations on words ([MSC+13]). Take for instance following classic example of this. Let $W2V(*)$ be the operation of converting word to its vector in embedding space and $V2W(*)$ be

the opposite operation of retrieving word by its vector. Next, consider following words: "king", "man", and "women". Then the following expression becomes true:

$$V2W(W2V(”king”) - W2V(”man”) + W2V(”women”)) \approx ”queen”.$$

That is, if we subtract the "man" part from the concept of "king", and add the "women" to it, we will get the vector that is close to the word "queen". This also implies that words can be meaningfully combined together by simple addition operation. If we take for instance top 5 nearest words to the vector of $W2V(”ukrainian”) + W2V(”river”)$ we will get concepts like *"Dnipro river", "Desna river", "ukrainian lake"* as the nearest candidates. These operations are possible due to the linear nature of the embedding space learned by the word2vec methods (see Figure 2).

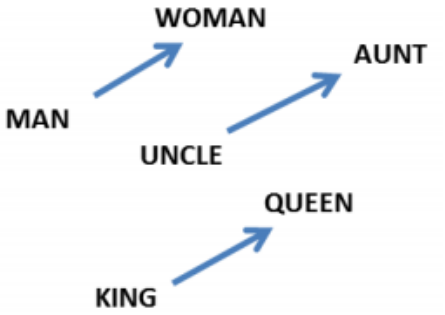


Figure 2. Linear relationships in embedding space (Figure comes from [Ola]).

The Skip-gram Model. There are two main word2vec architectures proposed by Milkov et al. in 2013 in [MCCD13]: the Continues Bag of Words model and the Skip-gram model. In this work we will consider the Skip-gram model since it has founded more wide use in the Unsupervised Machine Translation literature [1].

The main idea of the Skip-gram model is following. First we train a model that chooses some word from the sentence and predicts surrounding words based on it. That is, we predict the context for each word (refer to the Figure 3 to see the example). Then, we treat this task as an auxiliary one, and assume that word representations learned during this process are good representations for the semantical meaning of the words. Due to the distributional hypothesis it is reasonable to expect this.

The predictive model is a Feed Forward Neural Network with one linear hidden layer. The number of hidden units in this layer is equal to the desired embedding size. Weights that are between input and hidden layers are in fact our embedding matrix that

[1] To be more precise, the Skip-gram model is the state-of-the-art in Bilingual Lexicon Induction methods which we describe in the next subsection

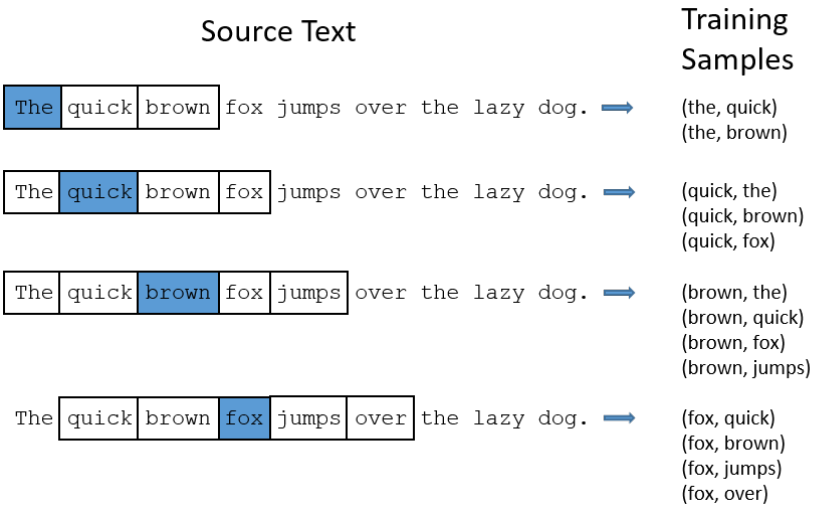


Figure 3. Formation of the training data for the Skip-gram model. The blue box highlights pivot word while context (surrounding) words are inside transparent boxes. (Figure comes from [McC]).

we will extract from the network ones it gets converged. So, we embed input word, resulting in real valued vector which we then linearly project to the vocabulary of words. On the output layer we, in simplest case, apply softmax function to get the probability distribution over the vocabulary of words. See Figure 4 for details.

We then (loosely speaking) correct the distribution based on the ground-truth context words by computing cross-entropy loss and applying backpropagation algorithm in order to perform gradient descent learning [2]. More recent method described in [BGJM17] incorporates subword level information in order to get embeddings for the words that were not seen during training, but employs the same skip-gram architecture as its core.

2.2 Unsupervised Bilingual Embedding Mapping

The first step of unsupervised machine translation is to apply one of the Unsupervised Bilingual Embedding Spaces Mapping techniques [ALA17, CLR+17, ALA18]). The goal of these methods is to obtain en embedding space where word vectors of the source language are close to the vectors of the analogues words in the target language. This is done by transforming the embedding matrix of one language to match the matrix of another language. The transformation here is a linear operation and usually even

[2] In reality, the procedure is more complex, and includes negative sampling loss instead of naive cross-entropy while some advanced techniques like subsampling of frequent words are used.

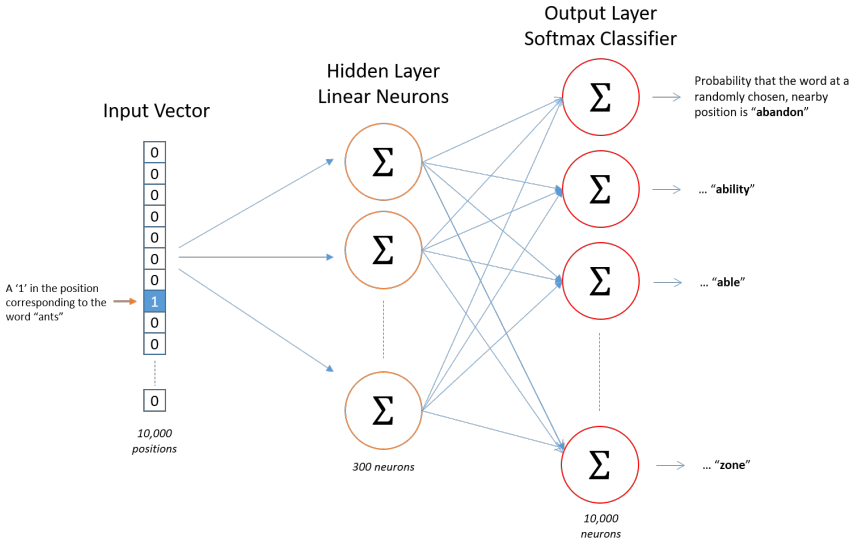


Figure 4. Skip-gram model architecture (Figure comes from [McC]).

includes orthogonality constrain, which means that only matrix rotation allowed as a part of transformation (see Figure 5 for details). The first step of bilingual mapping is

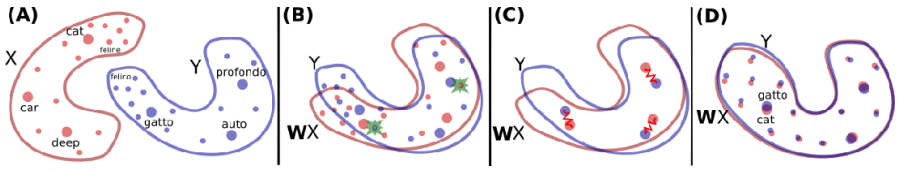


Figure 5. Bilingual mapping of the embedding spaces. We learn transformation matrix in order to iteratively obtain joint bilingual embedding space for 2 languages (Figure comes from [CLR+17]).

learning word vector spaces for both languages. This is usually done with the Skip-gram word embedding model. This model produces word semantically rich embedding spaces for words with linear relationships between elements inside the space.

2.3 Compositional and Non-Compositional Phrases

If the meaning of a phrase can be derived from its individual words, that phrase is called compositional. Otherwise, we call the phrase non-compositional. Examples of non-compositional phrases include idiomatic phrases like "couch potato", "state of the art", or named entities like "star wars" or "hey jude" (here and later throughout the thesis we

assume lowercasing for simplicity since it does not hurt our goal which is to demonstrate non-compositional nature of some phrases). In the case of "couch potato", individual words "couch" and "potato" have nothing to do with the result phrase meaning ("couch potato" is the name for the person who does not work-out and mostly spends her time relaxing at home).

3 Defining Requirements for Phrase Representations in the Context of Unsupervised Machine Translation

This section introduces 3 requirements to phrase representations suitable for Unsupervised Machine Translation.

3.1 Requirement 1: Ability to Embed Words and Arbitrary Phrases into the Same Semantically Rich Vector Space

Definition. Phrase representations for Phrase-based Unsupervised Neural Machine Translation have to be based on phrase embedding model that allows to get a semantically rich vector for an arbitrary phrase or word.

Motivation. Word-based machine translation methods rely on semantically rich word embeddings. Bilingual mapping methods and additional neural components of unsupervised machine translation methods assume that. Therefore, Phrase-based Machine Translation will also require semantically rich embedding space of words and phrases. Indeed, word embedding is the well-established method for converting text into fixed size vectors of real numbers that are then used as an input to the neural and other algorithms. Another important property of the word embedding models is that they (assuming large enough vocabulary) cover almost all words that occur in the corpus. Methods that incorporate character ngrams or subword level information ([BGJM17]) help with that. Consequently, we expect analogues property from the phrase embeddings as well.

3.2 Requirement 2: Ability to Apply Unsupervised Bilingual Embedding Mapping Methods

Definition. Phrase representations for Phrase-based Unsupervised Neural Machine Translation have to be suitable for unsupervised bilingual word embedding mapping methods.

Motivation. Bilingual Word Embedding Mapping Methods are the core necessary step of all of the current unsupervised machine translation approaches. There is no published work that would offer Unsupervised Machine Translation without this preliminary step. The mapping requires token embeddings for both languages which we assume to be the case if the requirement 1 is satisfied. However, in order for all the Unsupervised Bilingual Mapping Methods to work correctly, as were shown in [ALA18], these embedding spaces have to follow analogues linear regularities between elements. Assume that embedding space for the source language was learned with some technique that resulted in non-linear relationships between elements. Then consider another embedding space for the target language with another non-linear relationships. The mapping between these

two spaces has to be non-linear transformation, but there are no unsupervised methods that allow to learn such kind of non-linear function from our knowledge. Ability to do linear transformation is the main underlying assumption behind bilingual embedding mapping methods; therefore, vector spaces on both sides has to follow analogues linear relationships.

3.3 Requirement 3: Ability to Detect Non-compositional Phrases without Supervision

Definition. Phrase representations for Phrase-based Unsupervised Neural Machine Translation have to be suitable for non-compositional phrase detection methods

Motivation. Words have meanings that are specific to them and often can be interpreted independently on the context. Therefore, working with phrases introduces issue of non-compositional phrases. Some word combinations result in meaning that can not be derived from individual words meaning. In the Phrase-based Machine Translation we might want to translate a phrase word by word, but sometimes we might want translate a phrase as a whole. On the one hand, like in case of non-compositional phrase "couch potato", multiword expressions should be translated as a whole. On the other hand, some expressions like named entities, should not be translated at all, but rather left as they are. Since Unsupervised Translation relies on bilingual mapping, "Missouri river" (longest river in USA) can for instance be translated as "Gauja river" (biggest river in Estonia) and "Kersti Kaljulaid" (the president of Estonia) can become "Donald Trump" (the president of the USA), which we do not want to happen (these names might be semantic analogues to each other between embedding spaces). In both cases we want to detect these multiword expressions, so that we need phrase representations to be suitable for the non-compositionality detection.

4 Unsupervised Framework for Compositional Phrase Modeling

In this section we define and evaluate Unsupervised Framework for Compositional Phrase Modeling.

4.1 Non-compositional Model, Its Power and Limitations

In the previous section we defined requirements that has to be met in order for phrase representations to be suitable for Phrase-based Unsupervised Machine Translation. In this subsection we describe the Non-compositional Model as the solution that almost perfectly meets all defined requirements. Then we analyze this approach and show that it can not be effectively used in practice. In the following subsections we show how to overcome these limitations.

One solution towards phrase representations would be to concatenate all phrases with underscore (e.g. *"this_is_a_phrase"*) and treat them as individual vocabulary units in the Skip-gram model which we could train on enormous amount of monolingual data. We call this approach Non-compositional Modeling and it has the following nice properties that allow to satisfy to the defined requirements:

1) We directly learn a unified embeddings space for words and phrases with the Skip-gram model. Since the Skip-gram model is shown to produce semantically rich embedding space, the Requirement 1 is addressed.

2) Result phrase embeddings will not differ (in terms of their properties) from word embeddings. Word embeddings are learned as individual vocabulary units and satisfy to all assumptions of bilingual mapping methods; therefore, phrase embeddings learned this way would also do. Therefore, Requirement 2 is addressed.

3) Methods that address Requirement 3 (such as [YFH15]) are in fact designed to be applied to the phrase embeddings of this type.

Despite its high suitability, this approach has serious limitations. Firstly, it is very memory intensive because it is infeasible to learn and store vocabulary of all phrases that can occur in the corpus. The size of the vocabulary grows almost exponentially over the length of the phrase, and thus vocabularies of that order of magnitude do not fit into the computer memory in most cases. Secondly, there is a data sparsity problem since some (even two-word) phrases occur rarely even in the very large corpus (it makes learning embedding vectors hard for some phrases).

Following subsections present the unsupervised framework for compositional phrase modeling that overcomes this limitations while holding advantages of the non-compositional modeling approach.

4.2 Framework Definition

In this subsection, we define unsupervised framework for compositional phrase modeling. The core step of the framework was used as an intermediate step in solving other tasks in [MSY+14] and [YFH15], but was not considered neither in context of our task nor as a standalone method. We thus treat and evaluate it in the task-independent fashion and show that the framework avoids limitations of non-compositional model and satisfies to all constrains that phrase-based unsupervised machine translation implies on phrase representations.

Authors of [MSC+13] made an observation that simple addition of word vectors (learned with the Skip-gram model) results in meaningful phrase vectors. The idea to combine word embeddings to get a phrase embedding is a good fit for our needs, but addition operation while effective, may not be optimal way to do this combination. This leads us to the idea of applying machine learning techniques to learn the optimal parametrized function for combining words. We first learn vectors for reasonable amount of phrases as single-token vocabulary entries (e.g. "scientific_thesis") as with non-compositional modeling. At the same time (as a part of the same training procedure), we learn embeddings for individual words ("scientific" and "thesis" separately). Finally, we train regression model to predict phrase vectors from their word vectors. We assume that the model will learn the function that captures the pattern of combining word vectors in order to generate phrase vectors. Since we can obtain vectors for arbitrary words, we thus can estimate vectors for arbitrary phrases by combining word vectors with the learned function. Remaining text of the subsection define the framework in step by step fashion.

Step 1: obtain training data for non-compositional modeling. In order to compute vectors for words and subset of phrases (we treat phrases as single-unit vocabulary entries at this point) we first need to get a monolingual corpus and do some preprocessing like lowercasing / truecasing and tokenization.

Step 2: extract phrase candidates. Since we can not learn vectors for all phrases in the corpus, we have to decide which phrases to use to learn vectors for. One option is to randomly glue desired number of phrases together while other options include only gluing phrases that belong to some specific set of desired phrases. The set can be formed by scoring all phrases from the corpus by some criterion and then taking top N phrases based on their scores. Here we provide non-comprehensive list of criterion that can serve to the purpose of gluing two-word phrases: Likelihood ratio, Raw frequency, Poisson Stirling criterion, Chi square score, Dice score, Jaccard measure etc. We refer author to external literature for additional information on this topic (e.g. [MRS08]). Concrete metric should be task specific or empirically chosen.

Step 3: glue phrases. At this step we simply go through the corpus from Step 1 and glue some words together based on the phrases set from the Step 2.

Step 4: train word embedding model. At this step we train (say) the Skip-gram model on words and phrases corpus from Step 3. This way we get semantic vectors for

words and some phrases.

Step 4: obtain training data for compositional modeling. At this step we extract phrase vectors for phrases that we glued at the Step 3. Then we extract word vectors for words that are used to compose these phrases. The dataset then consists of following pairs of entities: sequence of the word vectors as an input, and phrase vector as the target.

For instance, one training example for two-word phrase might look like following:

$$[W2V(\text{"}hello\text{"}), W2V(\text{"}world\text{"})] \rightarrow W2V(\text{"}hello\_world\text{"})$$

Step 5: train compositional model on the data from the Step 4. At this point we use the dataset from previous step to teach the model to compose word vectors in a way that phrase vector is produced

This framework allows to get a model that predicts vectors for compositional and non-compositional phrases as if they were learned as the single-token units as a part of the vocabulary. We make our implementation of the framework available as an open source project [3].

4.3 Experiment Results

This subsection describes experiments we conducted with the framework and presents evaluation results. Concretely, we focused on the predictive ability of the different variants of the compositional models that we train as a part of the Step 5 of the framework. Following steps describe concrete decisions we made as a part of our implementation of the Unsupervised Compositional Phrase Modeling Framework.

Step 1: obtain training data for non-compositional modeling. We used first 1 billion bytes of English Wikipedia as our training data. The data contains 124,301,826 lowercased tokens.

Step 2: extract phrase candidates. We only glued phrases that belong to some specific set of desired phrases. The set was formed by scoring all phrases from the corpus by likelihood ratio criterion and then taking top 500,000 phrases based on their scores.

Step 3: glue phrases. At this step we simply went through the corpus from Step 1 and glued some words together based on the phrases set from the Step 2.

Step 4: apply skip-gram model. At this step we trained Skip-gram model on words and phrases corpus from Step 3. This way we got semantic real valued vectors for words and some phrases. We trained the system using *fasttext* framework ([BGJM17]) for 6 epochs with default parameters expect for the embedding size which we set to 100.

Step 4: obtain training data for compositional modeling. The result dataset size was about 600,000 training examples.

[3] `https://github.com/maxdel/bigram_embedder`

Step 5: train compositional model on the data from the Step 4. At this point we used the dataset from previous step to teach the model to compose word vectors in a way that phrase vector is produced.

Let *w1* be the vector of the first word and *w2* the vector of the second. Let also *p* be the result vector of the phrase and D be the dimension of the w1, w2, and p. The types of the models that we implemented and trained[4] are following.

- Simple addition *(AddSimple)*:

$$p = w1 + w2$$

- Addition with attention weights *(AddAtt)*:

$$p = a1 * w1 + a2 * w2$$

  where a1 and a2 are scalars that are learned by first concatenating the word vectors, and then projecting result into two dimensions.

- Dimwise addition with attention weights *(AddAttDimwise)*:

$$p = a1 * w1 + a2 * w2$$

  where a1 and a2 are vectors of the size D that are learned by first concatenating the word vectors, and then projecting result into the D dimensions. Therefore, we add two vectors with weight assigned to each dimension.

- Neural Network with one linear layer *(Linear)*:

$$W2 * (W1 * [w1, w2])$$

  where W1 and W2 are parameters matrices and [w1, w2] means concatenation

- Neural Network with dense ReLU layer and linear layer *(NonLinear)*:

$$W2 * ReLU(W1 * [w1, w2])$$

- Multilayer Neural Network *(MultilNonLinear)*: the same as the previous one, but with two more nonlinear layers. The sizes of hidden layers are 170, 130, and 100.

- Long Short Term Memory network *(LSTM)*: last timestep is used as phrase representation ([HS97]).

[4] *AddSimple* model does not require training while attention weights for *AddAtt* and *AddAttDimwise* are learned from data

The data was split into training, development, and test sets. The test set size was 2400 examples, development test size was 2000 examples, and the remaining data formed the training set. Training set was used to train the models, development set was used to tune models hyperparameters, and test set was used to perform final models comparison.

Smooth l1 loss was used in order to train all the models:

$$\text{loss}(x, y) = \frac{1}{n} \sum_i z_i$$

where

$$z_i = \begin{cases} 0.5(x_i - y_i)^2, & \text{if } |x_i - y_i| < 1 \\ |x_i - y_i| - 0.5, & \text{otherwise} \end{cases}$$

We choose this loss because it is more tolerant to outliers, which is desired property as we will see later in the Section 5.3.

Evaluation. In order to get interpretable accuracy scores for models comparison we first run our trained regression models in the inference mode to predict phrase vectors for the test set. Then we retrieve the top N (N is one of {1,3,5,7,10,100}) closest points in the embedding space, and check if the ground truth phrase belongs to this set. If the phrase is not in the topN, we count it as an error. Lastly we divide number of non-error examples by the size of the training set to obtain accuracy score. Table 1 shows accuracy scores across various models across various topN values.

Table 1. Accuracy results of explicit evaluation of compositional models

Model	**top1**	**top3**	**top5**	**top7**	**top10**	**top100**
AddSimple	0.35	0.81	0.88	0.91	0.94	0.95
AddAtt	0.37	0.65	0.74	0.78	0.84	0.95
AddAttDimwise	0.38	0.66	0.75	0.79	0.84	0.95
Linear	0.71	0.85	0.88	0.90	0.92	0.97
NonLinear	0.62	0.75	0.80	0.82	0.85	0.93
MultiNonLinear	0.69	0.83	0.87	0.88	0.91	0.97
LSTM	**0.73**	**0.88**	**0.92**	**0.94**	**0.95**	**0.98**

Analysis. As we can see from the Table 1, simple summation baseline shows decent performance in compositional phrase vector modeling. This interesting result was explained by the authors of the Skip-gram model ([MSC+13]). They show that addition of two token vectors approximately equivalent to the AND operation between their distributions over context words (we predict context / surrounding words with the Skip-gram model). This means that the result token vector will be equivalent to the token (phrase or word) that shares the same context with the input token vectors. Despite

good performance of the simple addition function, we observe drop in performance for attentional analogues. This might be due to the fact that it is sometimes hard to predict these attention weights from the words themselves, since the Skip-gram embeddings does not really contain much of Part of Speech (POS) information (e.g. words like "go", "goes", "going", "went" are grouped together despite being having different POS tags) while this information is what was needed to achieve good results in [MSY+14]. Among neural architectures, the LSTM network was able to outperform simple sum operation. It might be due to the separate gates it uses for memorizing the important (in context of the future phrase) semantic part of the word, forgetting redundant dimensions, and updating the first word with some information from the second word. We conducted experiments on phrases with length up to two words for simplicity.

This experiments show that LSTM network is a powerful tool for predicting compositional phrases while simple sum baseline remains a strong alternative.

5 Satisfying to Requirements with Compositional Phrase Modeling

5.1 Satisfying to Requirement 1: Predicting Single-token Phrase Embedding Vectors

The first requirement states that phrase representations for Phrase-based Unsupervised Neural Machine Translation have to be based on phrase embedding model that allows to get a semantically rich vectors for phrase or word. Compositional phrase modeling framework is based on the Skip-gram model which produces semantically rich embedding spaces. Phrase vectors are learned together with the word vectors so that we get joint embedding space for both phrases and words.

The requirement also states that one should be able to get representation for an arbitrary phrase. Compositional modeling framework aims to learn the function that produces phrase embedding based on its component word embeddings which means we can get an embedding for arbitrary phrase as long as we have embedding of its words (which we do have due to [BGJM17]).

Lastly, lets explore the question of whether these predicted phrase vectors are semantically rich enough. In the previous section we provided experimental results of direct evaluation of the predictive power of compositional function, but this does not tell how well these vectors perform on some external task. Authors of [MSY+14] performed such evaluation on phrase similarity dataset.

The dataset consists of phrase pairs of 3 types: Noun-Noun (NN-NN), Adjective-Noun (JJ-NN), and Verb-Noun (Vb-NN). Table 2 shows results for various compositional models.

Therefore, the work done in [MSY+14] strengthen our claims that phrase vectors learned with Compositional Modeling Framework are semantically rich and thus satisfy to the first requirement of phrase representations for phrase-based machine translation. Moreover, we evaluated the performance of our LSTM model on this dataset and got correlation scores of 0.49 for JJ-NN, 0.48 for NN-NN, and 0.39 for VB-NN thus achieving new state-of-the-are result for Adjective-Noun and Noun-Noun phrase pairs.

5.2 Satisfying to Requirement 2: Using in Bilingual Mapping

The second requirement states that phrase embeddings for Phrase-based Unsupervised Neural Machine Translation have to be suitable for unsupervised bilingual word embedding mapping methods. The task of studying such embeddings is novel from our knowledge. The authors of [CLR+17] touched this question a little, but their considered sentence embeddings and bag of embeddings approach. This however also falls under Compositional Modeling Framework we defined and is equivalent to the addition

	JJ-NN	NN-NN	VB-NN
Corpus	0.380	0.449	0.215
Add-smp	0.457	0.460	0.406
Add	0.335	0.304	0.338
Fulladd	0.359	0.359	0.366
RNN	0.364	0.360	0.367
Relfunc-add	0.440	0.455	0.388
Relfunc-cadd	0.419	0.445	0.413
Relfunc	**0.469***	**0.481***	**0.430***
Fulllex	0.322	0.160	0.222
Upper-bound	0.539	0.490	0.505

Table 2. Evidence of the semantical richness of the compositional phrase embeddings. *Corpus* corresponds to single-token embeddings, *Add-smp* to the simple addition, *Add* to the attentional addition, and *Fulladd* to dimwise addition. RNN means Recursive Neural Network (as opposite Recurrent NN) and other functions rely on additional supervised information (and thus are not useful in fully unsupervised settings). (Table comes from [MSY+14]).

composition function discussed in this work.

Addition (or bag of embeddings) is a mapping to the same space which holds linear properties. However, it is easy to show that more complex compositional models (like LSTM) presented here also does not break the linearity property that has to be met in order for bilingual mapping methods to work. These models mimic phrase vectors as if they were learned as single-token vocabulary entries together with words. Learning tokens as single-unit vocabulary entries is the what is currently done for words in Word-based Unsupervised Machine Translation and what is allows for Unsupervised Bilingual Mapping to Work. Moreover, as we showed in the previous section, using LSTM approach provides phrase vector that are closer to single-unit ones and thus are better suited for Bilingual Mapping.

Concretely, in order to perform Unsupervised Bilingual Mapping of the phrase vectors learned with the help of compositional modeling framework we do the following. First, we learn joint embedding spaces for words and phrases for both languages (Steps 1 - 4). These embedding spaces hold all the assumptions of the bilingual mapping methods and thus we apply Bilingual Mapping methods such as proposed in [ALA17] or [CLR+17] to obtain the transformation matrix. Finally, in order to map phrase vector or word vector in source language, we multiply it with the transformation matrix and get the

resulting vector in target language embedding space. If we do not have a vector for the phrase, we apply compositional model to estimate it from component words and apply transformation. As a proof of concept, we performed bilingual mapping for the English-Estonian embedding spaces obtained as a part of our compositional framework that works on LSTM model (since it is shown to perform the best; see Table 1). Manual evaluation shows that the mapping is reasonable. Lets look at the Table 3 that some examples of English tokens that are close to the vectors that was obtained by unsupervised mapping of the Estonian phrase or word.

Table 3. Top 5 English Nearest Neighbors to Estonian tokens

"suur kass" (big cat)	**"hea vein" (good wine)**	**"hea masintõlge" (good machine translation)**
big cat	old wine	machine translation
cute cats	bad wine	to translate
cats	tasty drink	translator
the cat	great wine	translation service
pretty kitty	this wine	automatic translation

In "hea vein" (good wine) example, despite having opposite meaning, "bad wine" is among closest phrase vectors. This is due to the fact, that words "good" and "bad" are used in similar contexts (they just mostly affect sentiment) and thus are closed in the Skip-gram embedding space. Also, we can see that for phrases like "suur kass" (big cat) and "hea masintõlge" (good machine translation) nearest neighbours account for noun verb and are less sensitive to adjective which is in general desired behaviour.

In summary, Unsupervised Modeling Framewrok is a solution for Phrase Embeddings that efficiently satisfy to the underlying assumptions of the Bilingual Embedding Mapping methods.

5.3 Satisfying to Requirement 3: Detecting Non-compositional Phrases

Requirement 3 states that our representations has to be suitable for non-compositional phrases detection. An interesting result is that the same unsupervised compositional modeling framework can be used in order to detect non-compositionality.

Since the framework is good at predicting phrase vectors from its component words, it should fail at predicting phrases meaning of which can not be derived from the individual words meanings. This in turn is a definition of non-compositional phrases. Indeed, for phrase vectors that we have learned as a part of the Skip-gram model training we can also try to estimate the phrase vectors from its individual words. If the difference in vectors is big enough, we can conclude that the meaning can not be derived from word vectors, and thus the phrase is non-compositional. Similar observations were made by authors of [YFH15], where they also do outliers detection as a way to find multiword expressions.

We also make sure that our compositional model do not overfit to the data be performing early stopping (early stopping can be seen as another regularization technique in Machine Learning).

We evaluated phrases for non-compostionality with our Compositional Phrase Modeling Framework. Following tables (Table 4, Table 5, Table 6) show behaviour of the framework on non-compostional phrases. We provide three lists of top 5 nearest neighbours for each non-compositional phrase. Three lists correspond to three methods we used to obtain the vector for the phrase. In the first case we take a vector learned for the phrase as a single-token as a part of the Skip-gram model. In the second case we use simple addition of word vectors. In the third case we use LSTM model to predict phrase vector from its words.

Table 4. Nearest neighbours for the vector of **"james bond"**

Single-token	**AddSimple**	**LSTM**
james bond	bond james	bond james
bond film	bond	bond
bond films	bond and	bond and
bond movie	ward bond	ward bond
s bond	bond to	bond as

Table 5. Nearest neighbours for the vector of **"counter strike"**

Single-token	**AddSimple**	**LSTM**
counter strike	strike	strike
tactical shooter	counter	counter strike
counter	strike	strike action
action game	strike action	counter
counter terror	strike against	strike against

Table 6. Nearest neighbours for the vector of **"bay area"**

Single-token	**AddSimple**	**LSTM**
bay area	bay	island area
francisco area	area	bay
francisco bay	island area	south shore
diego area	bayshore	port area
miami area	port area	southeast shore

Tables above show that the difference in top neighbours between non-compositional phrases learned as a single-token and the ones predicted by compositional model (SimpleAdd or LSTM) is significant. This explains significant difference between phrase vectors obtained with these methods. Table 4 explores the non-compositional phrase "james bond" where single token approach captures that this phrase is a movie character name while compositional models fail to do so. Phrase "counter strike" (Table 5) is usually used as a name of the one of the most popular computer games. We can see that compositional models show results that are close to the component words "counter" and "strike", and fail to capture the atomic nature of the phrase. Table 6 shows the same kind of behaviour for the "bay area", which is well-known part of the United States.

In conclusion, Unsupervised Modeling Frameworok can also be successfully used for non-compositionality detection.

6 Conclusions and Future Work

Phrase-based Unsupervised Machine Translation is a novel task and by the time of writing there is no published work that would propose the solution to it. However, we believe that this is the promising research direction and made as we believe an important step towards it. Concretely, we specified what makes a phrase representations effective in terms of Unsupervised Machine Translation, defined and evaluated the Unsupervised Compositional Modeling Framework for Phrases, and showed how to use this framework to satisfy to the proposed requirements. We showed that one can use Compositional Phrase Modeling Framework to obtain phrase and word embedding space which is semantically rich, regulated by linear relationships between elements (and thus suitable for Bilingual Embedding Mapping Methods), and suitable for detecting non-compositional phrases.

Future work includes studying an effect of other steps of our Unsupervised Framework for Compositional Phrase Modeling and eventually using Phrase Representations obtained with this framework as a part of the Phrase-based Unsupervised Machine Translation.

References

[ALA17] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[ALA18] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5012–5019, February 2018.

[ALAC17] Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. Unsupervised neural machine translation. *CoRR*, abs/1710.11041, 2017.

[Bal12] Pierre Baldi. Autoencoders, unsupervised learning, and deep architectures. In Isabelle Guyon, Gideon Dror, Vincent Lemaire, Graham Taylor, and Daniel Silver, editors, *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, volume 27 of *Proceedings of Machine Learning Research*, pages 37–49, Bellevue, Washington, USA, 02 Jul 2012. PMLR.

[BBCF16] Luisa Bentivogli, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. Neural versus phrase-based machine translation quality: a case study. *CoRR*, abs/1608.04631, 2016.

[BCB14] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

[BGJM17] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[CKS+17] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. *CoRR*, abs/1705.02364, 2017.

[CLR+17] Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *CoRR*, abs/1710.04087, 2017.

[GAG+17] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. *CoRR*, abs/1705.03122, 2017.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[KZS+15] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-thought vectors. *CoRR*, abs/1506.06726, 2015.

[LDR17] Guillaume Lample, Ludovic Denoyer, and Marc'Aurelio Ranzato. Unsupervised machine translation using monolingual corpora only. *CoRR*, abs/1711.00043, 2017.

[Lyn] Shane Lynn. Get busy with word embeddings – an introduction.

[McC] Chris McCormick. Word2vec tutorial - the skip-gram model.

[MCCD13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[MSC+13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013.

[MSY+14] Masayasu Muraoka, Sonse Shimaoka, Kazeto Yamamoto, Yotaro Watanabe, Naoaki Okazaki, and Kentaro Inui. Finding the best model among representative compositional models. In *PACLIC*, 2014.

[Ola] Christopher Olah. Deep learning, nlp, and representations.

[SH54] Zellig S. Harris. Distributional structure. 10:146–162, 08 1954.

[VBB+18] A. Vaswani, S. Bengio, E. Brevdo, F. Chollet, A. N. Gomez, S. Gouws, L. Jones, Ł. Kaiser, N. Kalchbrenner, N. Parmar, R. Sepassi, N. Shazeer, and J. Uszkoreit. Tensor2Tensor for Neural Machine Translation. *ArXiv e-prints*, March 2018.

[WSC+16] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus

Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016.

[YFH15] Majid Yazdani, Meghdad Farahmand, and James Henderson. Learning semantic composition to detect non-compositionality of multiword expressions. In *EMNLP*, 2015.

Appendix

I. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Maksym Del**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to:

   1.1 reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and

   1.2 make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

   of my thesis

   Towards Phrase-based Unsupervised Machine Translation: Phrase Representations

   supervised by Mark Fishel

2. I am aware of the fact that the author retains these rights.

3. I certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, 21.05.2018