NLP2 Project: *Compositionality decomposed*

Teaching assistant: *Verna Dankers* (vernadankers@gmail.com)

March 28, 2020

1 Introduction

Motivation This assignment addresses the principle of compositionality in languages and the ability of neural models commonly employed in *natural language processing* (NLP) to capture compositional languages. Compositionality refers to the principle that the meaning of a whole is a function of the meanings of the parts and of the way they are syntactically combined. A compositional language is defined by a theory of semantics and syntax, and the meaning of every grammatical sequence within that language ought to be derived from the application of valid compositions. In order for a language learner to succeed, we expect the learner to master the application of these compositions in a very generalised manner. For example, when being introduced to an artificial language containing rules `plie(ABC)` $\rightarrow$ `CBA` and `zor(ABC)` $\rightarrow$ `ABC`, one should have no issue processing `plie(zor(ABC))` if the nesting of rules is allowed by the language's syntax. Even processing `plie(zor(zor(zor(zor(ABC)))))` should be simple.

Neural models, however, frequently fail to generalise in a compositional manner when their training data did not contain sufficient evidence of a phenomenon: They struggle with generalising to input (or output) sequences that are longer than seen examples or to sequences that contain new combinations of tokens. They do not accurately recognise that synonymous input sequences should receive the same output. They fail to learn the meaning of a token or subsequence from few examples. To uncover this (un)compositional behaviour of neural models, empirical studies frequently employ artificial languages (tasks) that are designed to highlight desirable aspects of compositionality and consider testing conditions that are different from the conventional random separation of training and testing data. Hupkes et al. (2019) present five general compositionality tests that could be applied to a multitude of tasks:

1. *Systematicity*: tests whether models systematically recombine known parts and rules;
2. *Productivity*: tests whether models can extend their predictions beyond the length they have seen in the training data;
3. *Localism*: tests whether models' composition operations are local or global;
4. *Substitutivity*: tests whether models' predictions are robust to synonym substitutions;
5. *Overgeneralisation*: tests whether models favour rules or exceptions during training.

Assignment In this assignment, you will design a new artificial task using a *probabilistic context free grammmar* (PCFG). Your models will be trained on a sequence-to-sequence *machine translation* task where input sequences generated by the PCFG are translated into output sequences that represent their meanings. You will use the `OpenNMT-py`[1] library, designed for training models on machine translation tasks, to train two different types of models on your task. You will consider at least three testing conditions: one conventional testing condition with randomly split training and testing data

[1] Visit the library: `https://github.com/OpenNMT/OpenNMT-py`.

to compute the main task performance, and two compositionality tests that relate to one of the five principles of Hupkes et al. (2019). The artificial task, the models employed and the three testing conditions all serve as starting points for generating insights regarding the compositional behaviour of neural models, and you are expected to discuss these insights in a short paper.

Deliverables

1. Jupyter notebook, due April 24, 2020. The notebook should contain the entire pipeline from data generation to model training to the analysis conducted. Functions or classes are allowed to be defined in Python files externally, as long as the main functionality is listed in the notebook. We require you to train your models on GPUs through the Google Colab service.[2] To familiarise yourself with running `OpenNMT-py` from within a notebook, consider following a tutorial.[3]

2. Short paper, due April 24, 2020. The short paper should use the ACL conference's template[4] and contain four pages (references excluded). A suggested page distribution is as follows:

   (a) **Abstract**: summarise the research in a short piece of text that emphasises your contributions and findings (0.1 pages);

   (b) **Introduction**: introduce the reader to your research area, summarise your contributions and highlight the relevance of your research (0.5 pages);

   (c) **Related Work**: summarise research papers relevant for your work. Be brief, since this is a short paper (0.4 pages);

   (d) **Approach**: dependent on the particular project, this section should detail the tasks or models designed (1 page);

   (e) **Experiments and Results**: detail the precise experimental setup used and the numerical results your models achieved (1 page);

   (f) **Discussion**: interpret the results through additional analyses into your compositional task, the models and the three testing conditions (1 page).

   For the *Compositionality Decomposed* project the following components should be included:

   (a) A description of the artificial task in the 'Approach' section, including:

      i. The motivation for creating the language and the compositional phenomena it allows you to research;
      ii. The syntax through a PCFG;
      iii. The semantics through a list of interpretation rules;
      iv. The statistics of the machine translation task – e.g. the number of samples generated for training, testing and validating;
      v. The additional two compositionality tests.

   (b) A brief description of the two models you trained, along with the hyperparameters experimented with and the final experimental setup in the 'Experiments and Results' section.

   (c) Your numerical results in the 'Experiments and Results' section for:

      i. The main task performance;
      ii. The first compositionality test;
      iii. The second compositionality test.

[2] Visit Google Colab: `https://colab.research.google.com/`.
[3] For example the following tutorial:
`https://github.com/Parkchanjun/OpenNMT-Colab-Tutorial/blob/master/OpenNMT_Pytorch_Tutorial.ipynb`.
[4] Visit the template: `https://acl2020.org/calls/papers/#paper-submission-and-templates`.

(d) A discussion of the insights gained through the research. You are free to fill this section with analyses appropriate for your project (see 'Week 3' below). You are rewarded for the inclusion of thorough, creative and insightful analyses.

3. Poster presentation, due April 22, 2020. Compress the paper's content into a single-page poster that could be presented at a conference. Support the textual content through visual aids, such as tables and graphs that facilitate fast understanding of the paper's contributions and main results.

   For the *Compositionality Decomposed* project, include a clear description of the workings of the artificial task, mention the models trained and the compositionality tests conducted, and direct the reader's attention to insights regarding compositional behaviour of neural models.

2 Suggested Schedule

To stay on track, we recommend adhering to the following schedule. The lab sessions will start with a short presentation on the week's topic, containing references to the recommended reads to serve as inspiration and extend your knowledge of compositionality in neural networks.

2.1 Week 1: Data

Reading Read up on compositionality in neural networks and research papers that define new artificial tasks to research the ability of neural models to learn a language in a compositional manner. Go over the content of these papers to gather inspiration before defining a new artificial task.

1. Bowman et al. (2015) define a logical inference task (e.g. a sentence pair could be `(not` p_2`) and` p_6 `|` `not (`p_6 `or (`p_5 `or` p_3`))`);
2. Veldhoen et al. (2016) propose an arithmetic language that spells out mathematical expressions using brackets, numbers and operators (e.g. `( 5 + ( 2 - 3 ) )`);
3. Lake and Baroni (2018) present a sequence-to-sequence navigation task named SCAN that includes commands and modifiers that turn into series of actions (e.g. `turn left twice` becomes `LTURN LTURN`);
4. Liška et al. (2018) define lookup tables of functions that map bit strings to random other bit strings and can be composed (e.g. `c(010)` becomes `101`);
5. Hupkes et al. (2019) use a PCFG to define a sequence-to-sequence string edit task in which operators act on strings of characters (e.g. `remove_second A B C D , E F` becomes `A B C D`).

Coding Define an artificial language through a PCFG along with interpretation rules. Keep the broader research question in mind, so be mindful of the design of the language. Implement the PCFG and interpretation rules in the Jupyter notebook along with functionality to generate training, testing and validation sets. We recommend a dataset size between 25,000 and 100,000. Reuse the generated data for the compositionality test conditions.

Writing It is a good habit to write about the research along the way. At this point, the 'Introduction' and 'Related Work' sections could already be drafted after the literature review. One could write about the artificial task and the compositionality tests in the 'Approach' section, as well.

2.2 Week 2: Models

Reading Go over the functionality provided by `OpenNMT-py` (Klein et al., 2018). Vaswani et al. (2017) and Gehring et al. (2017) detail attention-based and convolution-based machine translation architectures that could serve as inspiration.

Coding Choose two types of sequence-to-sequence models to use and train them on the artificial task using the OpenNMT-py library. Select them with hypotheses about their behaviour in mind: A recurrent model might perform well for a task in which order is vital, while Transformer may have an advantage if the task requires modelling extremely long-range dependencies. Experiment with a limited set of hyperparameters for every model. Compute the numerical results for the three testing conditions. If needed, adapt the data if the task turns out to be too complicated or trivial.

Writing This week, the 'Experiments and Results' section is to be filled out. Extend the paper with an experimental setup detailing the models used and the hyperparameters experimented with and include the numerical results for the different testing conditions in a separate subsection.

2.3 Week 3: Analysis

Reading Now that the dataset, models and numerical results have been produced, it is time to perform analyses that offer insights about the compositional behaviour of neural models. The artificial task, the main task performance and the compositionality tests provide a framework to do so, but it is still up to the author to analyse and discuss this in the paper. Think of insightful and creative analyses and convey the conclusions in the paper through the 'Discussion' section and clarifying visualisations.

Return to the papers recommended in the first week for exemplary analyses or read the work of Dessì and Baroni (2019). In general, one could consider three types of analyses: They provide more detailed results based on the predictions emitted by the models (e.g. by computing results per input token). They analyse the internal workings of neural models (e.g. by examining attention distributions). They research the effect of a model's topology (e.g. by emphasising that certain hyperparameter combinations decrease the main task performance, but increase performance on other tests).

Coding Dependent on the types of analyses selected, they could be performed directly in the notebook (e.g. for analysing the models' outputs) or by using (or adapting) `OpenNMT-py` functionalities (e.g. for analysing embeddings or attention distributions). Indicate the analyses performed and the methods chosen to do so in the notebook.

Writing Write the 'Discussion' section of the paper. Summarise the conclusions of your analyses in text and ensure that these are supported by evidence presented in tables or graphs. Create an outline for the poster to be presented.

2.4 Week 4: Deliverables

Reading To finalise the project, consider reading the work of Yu et al. (2019) that has it all when it comes to defining a language, training models and conducting various analyses on the topic of generalisation and compositionality.

Coding Ensure that the notebook is understandable for the reviewers and that it details the entire pipeline from data generation to the analysis of results.

Writing Finish the paper and edit its content to fit into four pages. Additional analyses may be contained in the appendices, but the grade will be graded based on these first four pages. Include the paper's main points in the poster, supported by visual aids, such as tables and graphs that facilitate fast understanding of the artificial task, the most interesting results and the analyses conducted.

References

Samuel R Bowman, Christopher D Manning, and Christopher Potts. Tree-structured composition in neural networks without tree-structured architectures. In *Proceedings of the 2015th International Conference on Cognitive Computation: Integrating Neural and Symbolic Approaches-Volume 1583*, pages 37–42, 2015.

Roberto Dessì and Marco Baroni. Cnns found to jump around more skillfully than rnns: Compositional generalization in seq2seq convolutional networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3919–3923, 2019.

Jonas Gehring, Michael Auli, David Grangier, and Yann Dauphin. A convolutional encoder model for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 123–135, 2017.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. Compositionality decomposed: how do neural networks generalise? *CoRR/1908.08351*, 2019.

Guillaume Klein, Yoon Kim, Yuntian Deng, Vincent Nguyen, Jean Senellart, and Alexander M Rush. Opennmt: Neural machine translation toolkit. *Vol. 1: MT Researchers' Track*, page 177, 2018.

Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International Conference on Machine Learning*, pages 2873–2882, 2018.

Adam Liška, Germán Kruszewski, and Marco Baroni. Memorize or generalize? Searching for a compositional RNN in a haystack. In *Proceedings of AEGAP (FAIM Joint Workshop on Architectures and Evaluation for Generality, Autonomy and Progress in AI)*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Sara Veldhoen, Dieuwke Hupkes, Willem H Zuidema, et al. Diagnostic classifiers revealing how neural networks process hierarchical structure. In *CoCo@NIPS*, 2016.

Xiang Yu, Ngoc Thang Vu, and Jonas Kuhn. Learning the dyck language with attention-based seq2seq models. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 138–146, 2019.