# Experiments

Dieuwke Hupkes

## 1 Datasets

I define the following set of languages:

| **Name** | *Numeric leaves* | *Example* |
|---|---|---|
| $L_2$ | 2 | $(x_1\ op\ x_1)$ |
| $L_3$ | 3 | $((x_1\ op\ x_2)\ op\ x_3)$ |
| $L_4$ | 4 | $((x_1\ op\ x_2)\ op\ (x_3\ op\ x_4))$ |
| ... | | |

Where $x_i \in \{-19, 19\}$, and $op \in \{+, -\}$. The meaning $y$ of e sentences is the result of the arithmetic expression expressed by the languag. We restrict the languages to include only expressions such that $y \in \{-60, 60\}$.

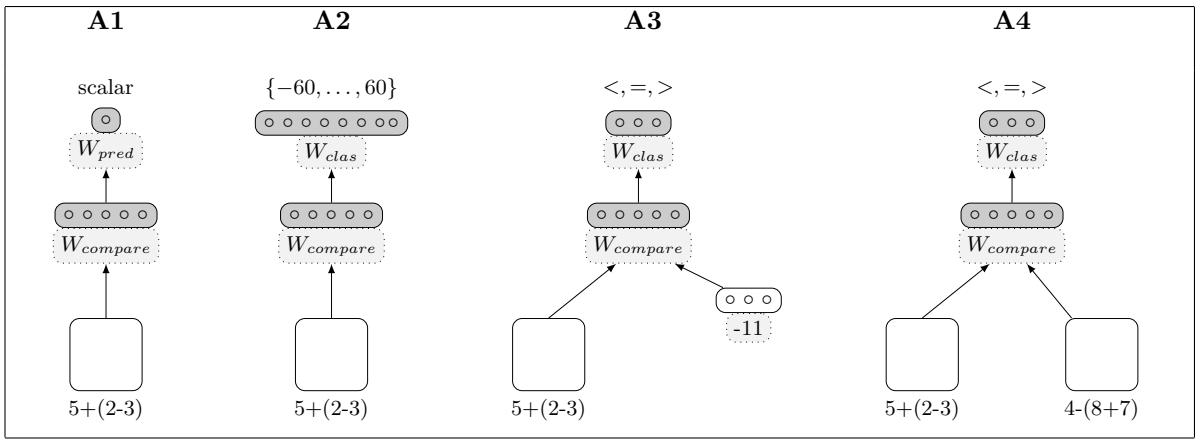
We define the following subsets of the languages defined above:

| **Name** | *Restriction* | *Example* | |
|---|---|---|---|
| $L_i+$ | $op == +$ | $(.(.(x_1 + x_2) + \dots x_i)$ | Structurally non ambiguous |
| $L_i-$ | $op == -$ | $(.(.(x_1 - x_2) - \dots x_i)$ | |
| $L_i$right | only right branching trees | $(.(.(x_1\ op\ x_2)\ op\ x_3)\ op\ \dots x_i)$ | Structurally non ambiguous |
| $L_i$left | only left branching trees | $(x_1\ op\ (x_2\ op\ (\dots\ op\ (x_{i-1}\ op\ x_i).).)$ | Structurally non ambiguous |

The datasets that the networks will be trained and tested on are (subsets of) unions of the languages described above.

## 2 Architectures

I use four different architectures (explanation?):



# 3 Experiments

I will start by running a sequence of experiments to determine if the networks can learn to compose the meaning of sentences from the structurally non ambiguous languages $L_2$, $L_3+$, $L_3left$ and $L_3right$. Depending on the results I will move on to more complicated languages In principle, I would like to do all (possible) combinations that can be made by combining elements from the following table,[1] starting with architectures A1 and A2 and then expanding to A3 and A4.

| **Network** | **Language** | **Architecture** | **Dimensionality** | **Initialisation** | **Embeddings** |
|---|---|---|---|---|---|
| SRN | $L_2$ | A1 | 10 | Random | fixed |
| GRU | $L_3+$ | A2 | 6 | Gray | trained |
| LSTM | | A3 | 2 | one-hot? | |
| | $L_3right$ | A4 | | | |
| | $L_3left$ | | | | |

# 4 Results

I ran a few run with Architecture 1, size_hidden = 20, size_compare=10, size_embeddings = 2 and language = L2. The trainingset contains 1800 sentences, validation set contains 200 sentences. Batchsize = 24. During most runs the prediction error (= the sum squared differences between the true outcome and the rounded scalar prediction of the network) on the validation set stays high for a while and then rapidly decreases to a value close to 0.

[1] Of course excluding non-sensical combinations, such as Gray encoding in two dimennions

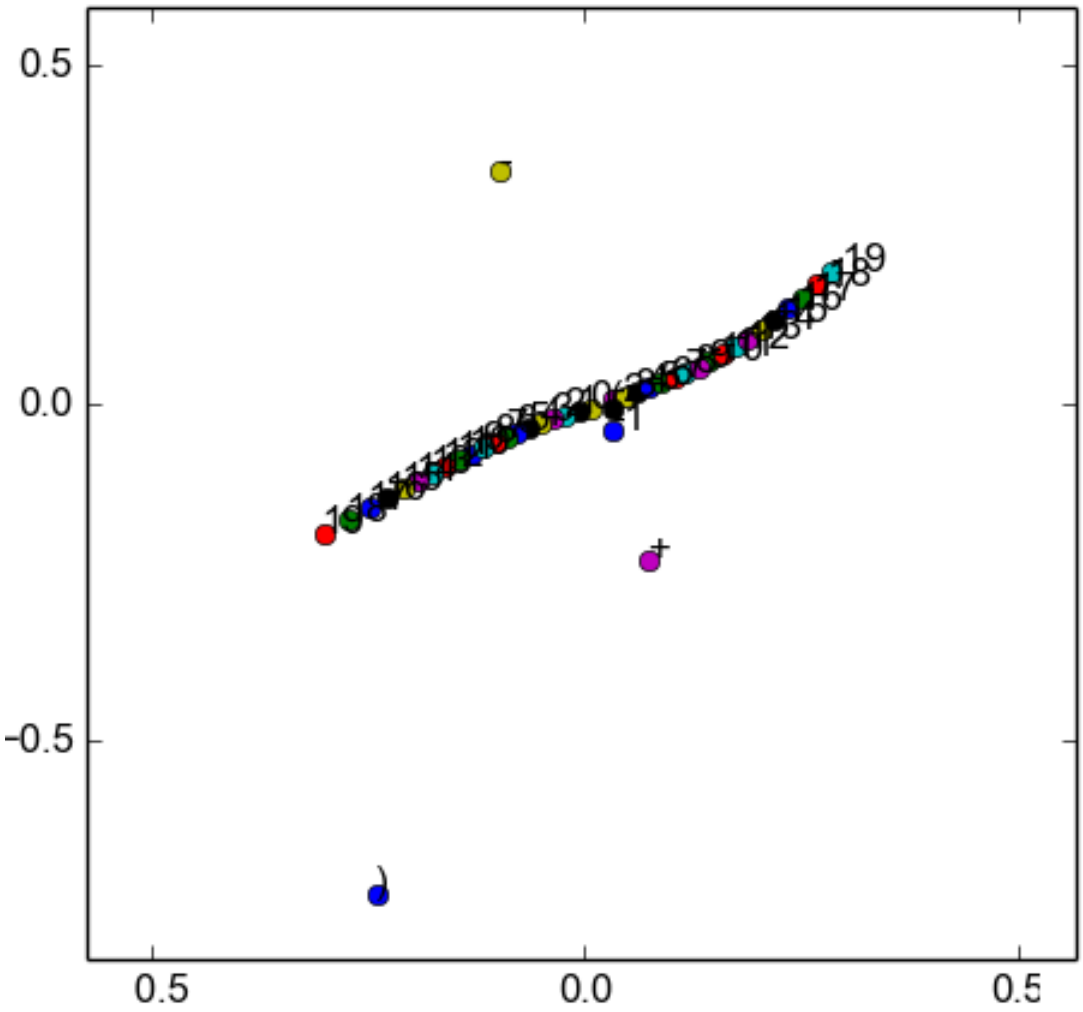



Figure 1: Learned embeddings for network trained on 1800 L2 sentences and tested on 200 different L2 sentences. After training, the mean squared prediction error on the 200 sentences in the validation set was 0.05

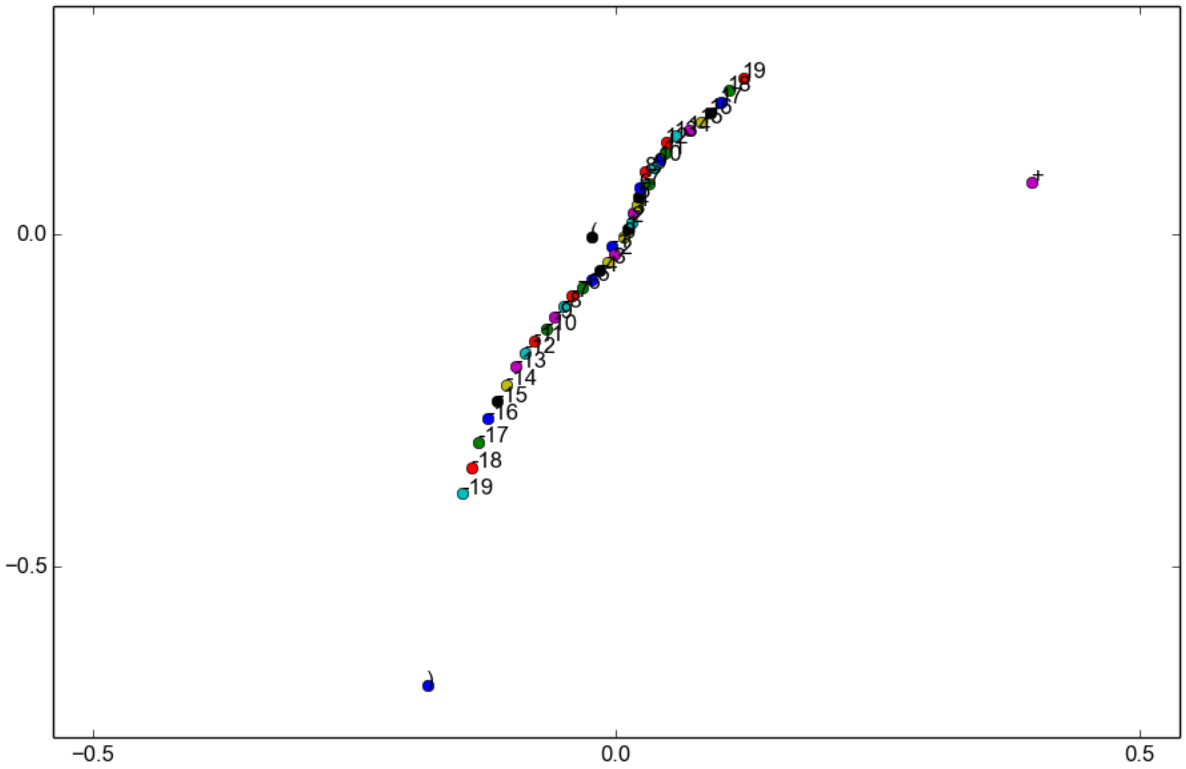



Figure 2: Learned embeddings for network trained on 1800 L2 sentences and tested on 200 different L2 sentences. After 1500 epochs, the mean squared prediction error on the 200 sentences in the validation set was 0.76