

# Evaluating neural network explanation methods using hybrid documents and morphosyntactic agreement (supplementary material)

Nina Poerner, Benjamin Roth & Hinrich Schütze  
Center for Information and Language Processing  
LMU Munich, Germany  
poerner@cis.lmu.de

## 1 Corpora and data preprocessing

The 20 newsgroups corpus (Lang, 1995) was downloaded using the Python `sklearn` package (Pedregosa et al., 2011), removing all headers, footers and quotes. The corpus contains 18,846 posts and comes with a training and test set. We randomly split the latter into a heldout and a test set.

For sentiment analysis we use the Pennsylvania subset of the 10th yelp dataset challenge<sup>1</sup>. It contains 206,338 reviews with 1 to 5 star ratings. 1 or 2 stars are mapped to “negative”, 4 or 5 stars to “positive”, 3 star reviews are discarded. We randomly split the data into training, heldout and test sets (90%/5%/5%). On both corpora, we use NLTK (Bird et al., 2009) for word and sentence tokenization. Words with a frequency rank above 50000 are mapped to *oov*. To create hybrid documents, we sentence-tokenize the test sets, shuffle, and then concatenate ten sentences at a time.

The manually annotated 20 newsgroups documents were obtained from Mohseni and Ragan (2018)<sup>2</sup>. The relevance ground truth consists of one list of lowercased word types per document. There are a number of mismatches between the ground truth and the documents (e.g., one list contains *rays* but its document only contains *x-rays*). This made some reverse engineering necessary: Given  $\mathbf{X}$  and its list, we add  $t$  to  $\text{gt}(\mathbf{X})$  if lower-cased  $x_t$  is a prefix or suffix of at least one word type in the list.

For the morphosyntactic agreement experiment, we use Linzen et al. (2016)<sup>3</sup>’s corpus of 1,577,211 English Wikipedia sentences with automatic morphosyntactic annotation<sup>3</sup>. We replicate the original dataset sizes (9% train, 1% heldout, 90% test). Like in the original corpus, words with a frequency rank above 10,000 are replaced by their part-of-speech tag.

## 2 Neural networks

Every neural network used in our paper is made up of a word embedding matrix, followed by a core layer, followed by a fully-connected layer with softmax activation.

In the hybrid document experiment, the  $|V| \times 300$  embedding matrix is initialized with GloVe embeddings (Pennington et al., 2014)<sup>4</sup>, which are fine-tuned during training. The core layer is a bidirectional Gated Recurrent Unit (GRU, Cho et al. (2014)), bidirectional Long-Short Term Memory Network (LSTM, Hochreiter and Schmidhuber (1997)), bidirectional Quasi-GRU or Quasi-LSTM (Bradbury et al., 2017), or a 1D Convolutional Neural Network (CNN) with global max

pooling (Collobert et al., 2011). In all cases, the core layer has a hidden size of 150 (bidirectional architectures: 75 per direction), for QRNNs and CNN, we use a kernel width of 5. For regularization, we use 50% dropout between layers and on hidden-to-hidden connections (GRU/LSTM only).

We minimize categorical crossentropy using Adam (Kingma and Ba, 2015), with learning rate 0.001,  $\beta_1 = 0.9$ ,  $\beta_2 = 0.999$  and batch size 8. Heldout accuracy is monitored; after two stagnant epochs, the learning rate is halved, and after 5 (yelp), resp. 25 (20 newsgroups), stagnant epochs, training is stopped and the model from the best epoch is stored. Final test set accuracies are .964/.954/.965/.959/.957 on yelp and .727/.716/.730/.735/.705 on 20 newsgroups (GRU/QGRU/LSTM/QLSTM/CNN).

In the morphosyntactic agreement experiment, the  $|V| \times 50$  embedding matrix is randomly initialized. All (Q)RNNs are unidirectional and have a hidden size of 50. QRNN kernel width is 5. The core layer is followed by a fully connected  $50 \times 2$  layer with softmax activation. We minimize categorical crossentropy using Adam (see above), with early stopping after 20 epochs based on heldout accuracy, and a batch size of 16. Final test set accuracies are .991/.985/.990/.986 (GRU/QGRU/LSTM/QLSTM). Contrary to Linzen et al. (2016), we do not train an ensemble.

### 2.1 GRU

$$\begin{aligned}\vec{h}_0 &= 0 \\ \vec{z}_t &= \sigma(\mathbf{V}_z \vec{e}_t + \mathbf{U}_z \vec{h}_{t-1} + \vec{b}_z) \\ \vec{r}_t &= \sigma(\mathbf{V}_r \vec{e}_t + \mathbf{U}_r \vec{h}_{t-1} + \vec{b}_r) \\ \vec{g}'_t &= \mathbf{V} \vec{e}_t + \mathbf{U}(\vec{r}_t \odot \vec{h}_{t-1}) + \vec{b} \\ \vec{g}_t &= \tanh(\vec{g}'_t) \\ \vec{h}_t &= \vec{z}_t \odot \vec{h}_{t-1} + (\vec{1} - \vec{z}_t) \odot \vec{g}_t\end{aligned}$$

### 2.2 QGRU

$$\begin{aligned}\mathbf{Z} &= \sigma(\mathbf{V}_z \star [0 \dots \vec{e}_1 \dots \vec{e}_T] + \vec{b}_z) \\ \mathbf{G}' &= \mathbf{V} \star [0 \dots \vec{e}_1 \dots \vec{e}_T] + \vec{b} \\ \mathbf{G} &= \tanh(\mathbf{G}') \\ \vec{h}_0 &= 0 \\ \vec{h}_t &= \vec{z}_t \odot \vec{h}_{t-1} + (1 - \vec{z}_t) \odot \vec{g}_t\end{aligned}$$

<sup>1</sup>[www.yelp.com/dataset\\_challenge](http://www.yelp.com/dataset_challenge)

<sup>2</sup><http://github.com/SinaMohseni/ML-Interpretability-Evaluation-Benchmark>

<sup>3</sup>[www.tallinzen.net/media/rnn\\_agreement/agr\\_50\\_mostcommon\\_10K.tsv.gz](http://www.tallinzen.net/media/rnn_agreement/agr_50_mostcommon_10K.tsv.gz)

<sup>4</sup><http://nlp.stanford.edu/data/glove.840B.300d.zip>

## 2.3 LSTM

$$\begin{aligned}
\vec{c}_0 &= \vec{h}_0 = 0 \\
\vec{i}_t &= \sigma(\mathbf{V}_i \vec{e}_t + \mathbf{U}_i \vec{h}_{t-1} + \vec{b}_i) \\
\vec{f}_t &= \sigma(\mathbf{V}_f \vec{e}_t + \mathbf{U}_f \vec{h}_{t-1} + \vec{b}_f) \\
\vec{o}_t &= \sigma(\mathbf{V}_o \vec{e}_t + \mathbf{U}_o \vec{h}_{t-1} + \vec{b}_o) \\
\vec{g}'_t &= \mathbf{V} \vec{e}_t + \mathbf{U} \vec{h}_{t-1} + \vec{b} \\
\vec{g}_t &= \tanh(\vec{g}'_t) \\
\vec{c}_t &= \vec{f}_t \odot \vec{c}_{t-1} + \vec{i}_t \odot \vec{g}_t \\
\vec{h}_t &= \vec{o}_t \odot \tanh(\vec{c}_t)
\end{aligned}$$

## 2.4 QLSTM

$$\begin{aligned}
\mathbf{I} &= \sigma(\mathbf{V}_i \star [0 \dots \vec{e}_1 \dots \vec{e}_T] + \vec{b}_i) \\
\mathbf{F} &= \sigma(\mathbf{V}_f \star [0 \dots \vec{e}_1 \dots \vec{e}_T] + \vec{b}_f) \\
\mathbf{O} &= \sigma(\mathbf{V}_o \star [0 \dots \vec{e}_1 \dots \vec{e}_T] + \vec{b}_o) \\
\mathbf{G}' &= \mathbf{V} \star [0 \dots \vec{e}_1 \dots \vec{e}_T] + \vec{b} \\
\mathbf{G} &= \tanh(\mathbf{G}') \\
\vec{h}_0 &= \vec{c}_0 = 0 \\
\vec{c}_t &= \vec{f}_t \odot \vec{c}_{t-1} + \vec{i}_t \odot \vec{g}_t \\
\vec{h}_t &= \vec{o}_t \odot \tanh(\vec{c}_t)
\end{aligned}$$

## 2.5 CNN

$$\begin{aligned}
\mathbf{G}' &= \mathbf{V} \star [0 \dots \vec{e}_1 \dots \vec{e}_T \dots 0] + \vec{b} \\
\mathbf{G} &= \text{relu}(\mathbf{G}') \\
h_d &= \max_t(g_{t,d})
\end{aligned}$$

## 3 RGB coding in examples

$$\begin{aligned}
\phi'(t, k, \mathbf{X}) &= \frac{\phi(t, k, \mathbf{X})}{\max_{t'}(1.1|\phi(t', k, \mathbf{X})|)} \\
R(t, k, \mathbf{X}) &= \phi'(t, k, \mathbf{X}) \mathbb{I}[\phi(t, k, \mathbf{X}) < 0] \\
G(t, k, \mathbf{X}) &= \phi'(t, k, \mathbf{X}) \mathbb{I}[\phi(t, k, \mathbf{X}) > 0] \\
B(t, k, \mathbf{X}) &= 0
\end{aligned}$$

## 4 Epsilon LRP and DeepLIFT

In the following, we assume that the hidden layer relevance vector  $R(\vec{h})$  (resp.  $R(\vec{h}_T)$ ) has been backpropagated by the upstream fully connected layer using equations from Sections 3.2 and 3.3 (main paper). DeepLIFT can be derived by replacing  $h, g, g', e, c$  with  $h - \bar{h}, g - \bar{g}, g' - \bar{g}', e - \bar{e}, c - \bar{c}$ .  $F$  is CNN / QRNN kernel width.

## 4.1 GRU

$$\begin{aligned}
R(g_{t,d}) &= R(h_{t,d}) \frac{g_{t,d} \cdot (1 - z_{t,d})}{h_{t,d} + \text{esign}(h_{t,d})} \\
R(e_{t,d}) &= \sum_{j=1}^{\dim(\vec{g}_t)} R(g_{t,j}) \frac{e_{t,d} \cdot v_{d,j}}{g'_{t,j} + \text{esign}(g'_{t,j})} \\
R(h_{t-1,d}) &= R(h_{t,d}) \frac{h_{t-1,d} \cdot z_{t,d}}{h_{t,d} + \text{esign}(h_{t,d})} \\
&\quad + \sum_{j=1}^{\dim(\vec{g}_t)} R(g_{t,j}) \frac{h_{t-1,d} \cdot r_{t,d} \cdot u_{d,j}}{g'_{t,j} + \text{esign}(g'_{t,j})}
\end{aligned}$$

## 4.2 QGRU

$$\begin{aligned}
R(g_{t,d}) &= R(h_{t,d}) \frac{g_{t,d} \cdot (1 - z_{t,d})}{h_{t,d} + \text{esign}(h_{t,d})} \\
R(h_{t-1,d}) &= R(h_{t,d}) \frac{h_{t-1,d} \cdot z_{t,d}}{h_{t,d} + \text{esign}(h_{t,d})} \\
R(e_{t,d}) &= \sum_{j=1}^{\dim(\vec{g}_t)} \sum_{k=0}^{F-1} R(g_{t+k,j}) \frac{e_{t,d} \cdot v_{k,d,j}}{g'_{t+k,j} + \text{esign}(g'_{t+k,j})}
\end{aligned}$$

## 4.3 LSTM

$$\begin{aligned}
R(c_{T+1,d}) &= 0 \\
R(c_{t,d}) &= R(h_{t,d}) \frac{\tanh(c_{t,d}) \cdot o_{t,d}}{h_{t,d} + \text{esign}(h_{t,d})} \\
&\quad + R(c_{t+1,d}) \frac{c_{t,d} \cdot f_{t+1,d}}{c_{t+1,d} + \text{esign}(c_{t+1,d})} \\
R(g_{t,d}) &= R(c_{t,d}) \frac{g_{t,d} \cdot i_{t,d}}{c_{t,d} + \text{esign}(c_{t,d})} \\
R(e_{t,d}) &= \sum_{j=1}^{\dim(\vec{g}_t)} R(g_{t,j}) \frac{e_{t,d} \cdot v_{d,j}}{g'_{t,j} + \text{esign}(g'_{t,j})} \\
R(h_{t-1,d}) &= \sum_{j=1}^{\dim(\vec{g}_t)} R(g_{t,j}) \frac{h_{t-1,d} \cdot u_{d,j}}{g'_{t,j} + \text{esign}(g'_{t,j})}
\end{aligned}$$

## 4.4 QLSTM

$$\begin{aligned}
R(c_{T+1,d}) &= 0 \\
R(c_{t,d}) &= R(h_{t,d}) \frac{\tanh(c_{t,d}) \cdot o_{t,d}}{h_{t,d} + \text{esign}(h_{t,d})} \\
&\quad + R(c_{t+1,d}) \frac{c_{t,d} \cdot f_{t+1,d}}{c_{t+1,d} + \text{esign}(c_{t+1,d})} \\
R(g_{t,d}) &= R(c_{t,d}) \frac{g_{t,d} \cdot i_{t,d}}{c_{t,d} + \text{esign}(c_{t,d})} \\
R(e_{t,d}) &= \sum_{j=1}^{\dim(\vec{g}_t)} \sum_{k=0}^{F-1} R(g_{t+k,j}) \frac{e_{t,d} \cdot v_{k,d,j}}{g'_{t+k,j} + \text{esign}(g'_{t+k,j})}
\end{aligned}$$

## 4.5 CNN

$$\begin{aligned}
F' &= \frac{F-1}{2} \\
R(g_{t,d}) &= R(h_d) \cdot \mathbb{I}[\arg\max_{t'}(g_{t',d}) = t] \\
R(e_{t,d}) &= \sum_{j=1}^{\dim(\vec{g})} \sum_{k=-F'}^{F'} R(g_{t+k,j}) \frac{e_{t,d} \cdot v_{k,d,j}}{g'_{t+k,j} + \text{esign}(g'_{t+k,j})}
\end{aligned}$$



















## References

- Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: Analyzing text with the natural language toolkit*. O'Reilly Media.
- James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. 2017. Quasi-recurrent neural networks. In *International Conference on Learning Representations*, Toulon, France.
- Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. In *Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar.
- Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.
- Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, San Diego, USA.
- Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *International Conference on Machine Learning*, pages 331–339, Tahoe City, USA.
- Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.
- Sina Mohseni and Eric D Ragan. 2018. A human-grounded evaluation benchmark for local explanations of machine learning. *CoRR*, abs/1801.05075.
- Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12(Oct):2825–2830.
- Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar.