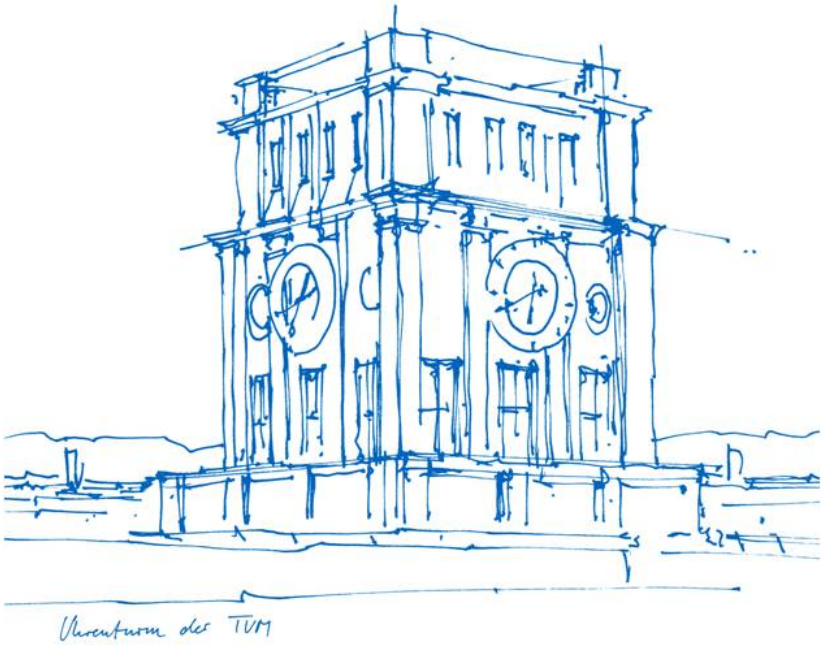
Exercises for Social Gaming and Social Computing (IN2241 + IN0040) – Introduction to Exercise 6



Exercise Content

Sheet Number	Exercise	Working Time
1	• Introduction to Python and Network Visualization	May 29 - June 11 (sic!)
2	• Centrality Measures	June 5 – June 11 (sic!)
3	• Finding Groups & Clustering Methods	June 12 – 18
4	• Predicting Social Tie Strength with Linear Regression	June 19 - 25
5	• Natural Language Processing: Hate Speech detection + Social Context Influence	June 26 – July 2
6	• Natural Language Processing: Modern Machine Learning Methods and Explainable AI	July 3 – July 9

Exercise Sheet 6 – BERT and XAI

Exercise Sheet 6 might look very long on paper but don‘t worry. The sheet just contains lots of text and in addition to that you can reuse your code from sheet 5.

Goals:

- Ability to use basic NLP: Trabsformer-Based Neural Networks with Word-Embeedings as input.
- Ability to make use of basic explainability methods (SHAP), understand limitations of ML-based NLP

Data set:

- hand-labeled tweets by Waseem & Hovy [1]. Each tweet is either labeled as 'sexism', 'none' or 'racism'.

The Data

- models.social_bilstm_model_waseemhovy.h5: trained and ready-to-use Bi-LSTM model with social information for comparison
- From exercise 5: src.evaluation.py: provides useful functions for you to use (F1-Score)
- From exercise 5: pickle_files.data_to_be_explained: provides social information about the authors
  - tweets_author_hate_score_avg.npy: your computed hate scores from sheet 5
- glove.6b.100d.txt: downloaded, pre-computed GloVe vectors for word embedding
- From exercise 5: tweets.csv: provides you with the labeled tweets from Waseem & Hovy

The Data

- models.social_bilstm_model_waseemhovy.h5: trained and ready-to-use Bi-LSTM model with social information for comparison
- src.evaluation.py: provides useful functions for you to use (F1-Score)
- src.explainability.py: provides useful functions for you to use and extend on, you will have to read and understand this code!
- pickle_files.data_to_be_explained: provides social information about the authors
  - tweets_author_hate_score_avg.npy: your computed hate scores from sheet 5
- glove.6b.100d.txt: downloaded, pre-computed GloVe vectors for word embedding
- tweets.csv: provides you with the labeled tweets from Waseem & Hovy

Task 6.1: Prepocessing

- Encode the labels etc: You can reuse your code from sheet 5. If you have not done sheet 5 read its task to understand the label encoding!

- We will use a trained NN (DistilBert): install the transformers libary

- We will use Shapley values for Explainable AI (XAI): install the shap 0.41.0 libary

2.3 BERT for Toxic speech classification

We will fine-tune to our downstream task BERT model [3], more specifically – distilled version of it DistilBERT.

[ ]:
```
# TODO: initialize tokenizer and model for DistilBERT or
# for other model of your preferences -- you are very welcome to try out␣
 ↪something different!

tokenizer = ##
model = ##
```

2.3.1 Training Batches Preparation

The same as in the previous tutorial, we will create our custom datasets and loaders to generate batches for training. However, we need to adapt it to transformers input:

1. Each dataset item should return `input_ids`, `attention_mask`, and `label`.
2. All should be `tensors`.
3. In the end, you need to apply `collate_fn` – that will pad all tensors in batches to the max_length (already implemented for you).

[ ]:
```
# TODO: create your CustomDataset

class CustomDataset(Dataset):
    def __init__(self, texts, labels, tokenizer):
        # TODO

    def __len__(self):
        # TODO

    def __getitem__(self, index):
        # TODO

# -------------- End of Solution ------------------
```

2.3.2 Training Loop

We are ready to train the model! You can reuse the code from the previous tutorial: 1. Define `optimizer` and `criterion` for a classiication task. 2. Use `train_lodaer` to sample batches for training. 3. Track validation loss using data from `val_loader`. 4. Achtung: now the items in batches have different structure!

[ ]:
```
%%time

# TODO: implement training loop

# you can play with different number of epochs and check the model's␣
↪performance!
num_epochs = 3

###
```

[ ]:
```
# TODO: Evaluate the model based on test_loader

###
```

2.4 SHAP Explanations

Now, we have a decent model for toxic speech detection. However, sometimes it can be not so clear why some sample is considered toxic or not. We can try to explaine the model's decision! For this, we utilize SHAP.

TODO: how can we understand if some model is better or not? You can pick the misclassified samples for each model and compare the explanations. Can you explaine why the model did mistakes? Are these mistakes the same or not? Try out other ways how to use SHAP here. Maybe, other ways of explanations can be useful more?

TODO: Write your observations here

Concluding questions: * Do the hate scores perform as expected in our model? * If not, can you come up with a possible explanation for that even though the models with social scores performed better? * What does that tell you about applicability of neural networks and their trustworthiness?

TODO: Write your answers here

Task 6.3: Shap

One issue with Neural Networks is that it is hard to explain why a certain outcome occurs. To be able to understand what is actually happening in the Neural Network, many researchers are working in the field of explainability, because many models are seen as black boxes whose "decisions" cannot be comprehended.

One potential module that we can use for explainability is the Shap module [9]. It introduces a game theoretic approach that can be used to explain the output of any Neural Network. We are using it to be able to see what inputs are important for our Bi-LSTM Model. In our example we use a DeepExplainer since our model uses a Neural Network. The return values are the so called Shapley [10] values. They describe how each input feature contributes to a certain output label. To make the features readable we need to translate our tokenized integer sequence back to words.

- To understand the idea of the Shapley values take a look at https://christophm.github.io/interpretable-ml-book/shapley.html

Excursion: Transformers

To understand the idea of Transformers take a look at
http://jalammar.github.io/illustrated-transformer/

Language Models

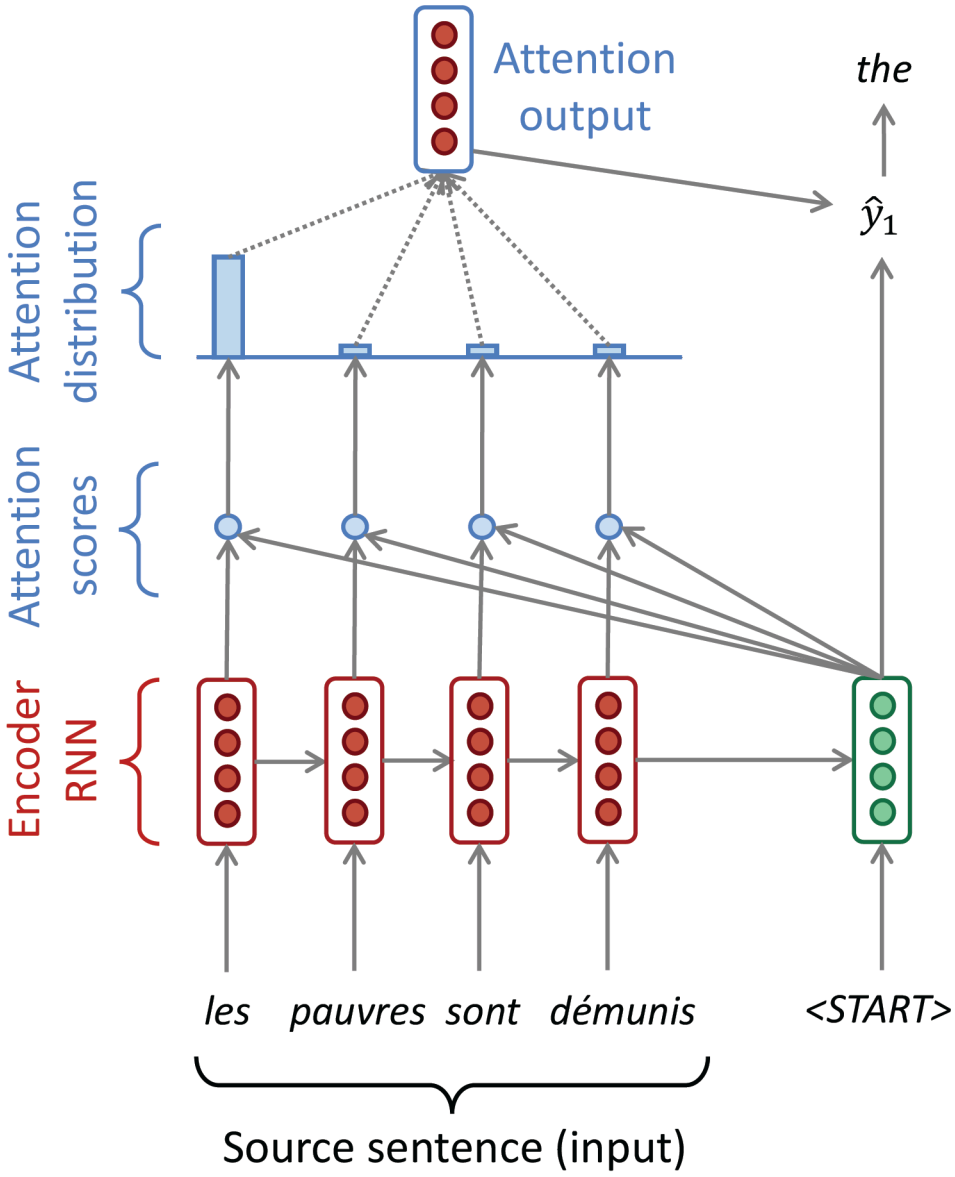
- task: predicting next word

$$P(\boldsymbol{x}^{(t+1)} | \ \boldsymbol{x}^{(t)}, \ldots, \boldsymbol{x}^{(1)})$$

the students opened their …				
$x^{(1)}$	$x^{(2)}$	$x^{(3)}$	$x^{(4)}$	$x^{(5)}$

- good indicator of overall progress in NLP.

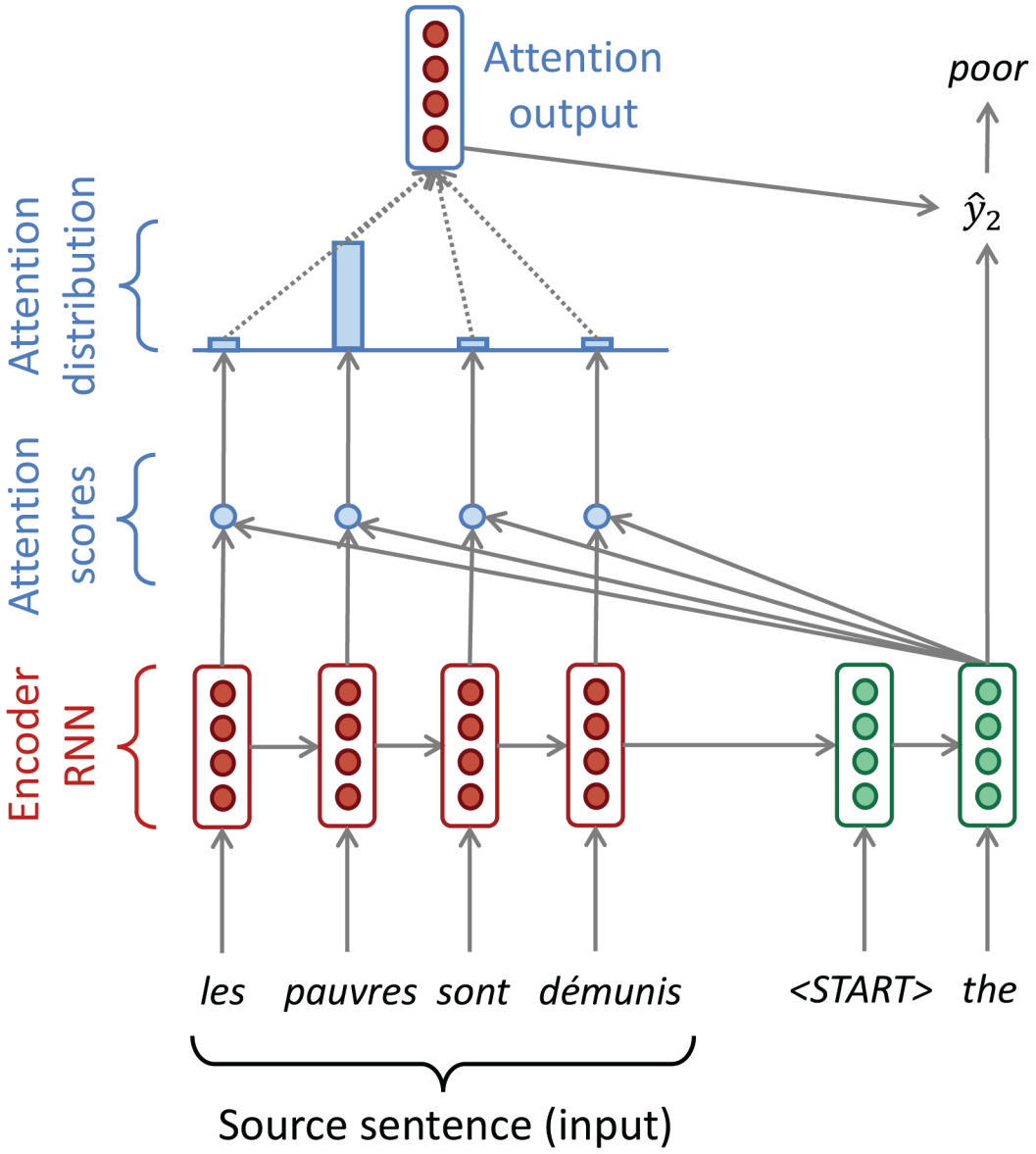
Dot-Product Attention [9]



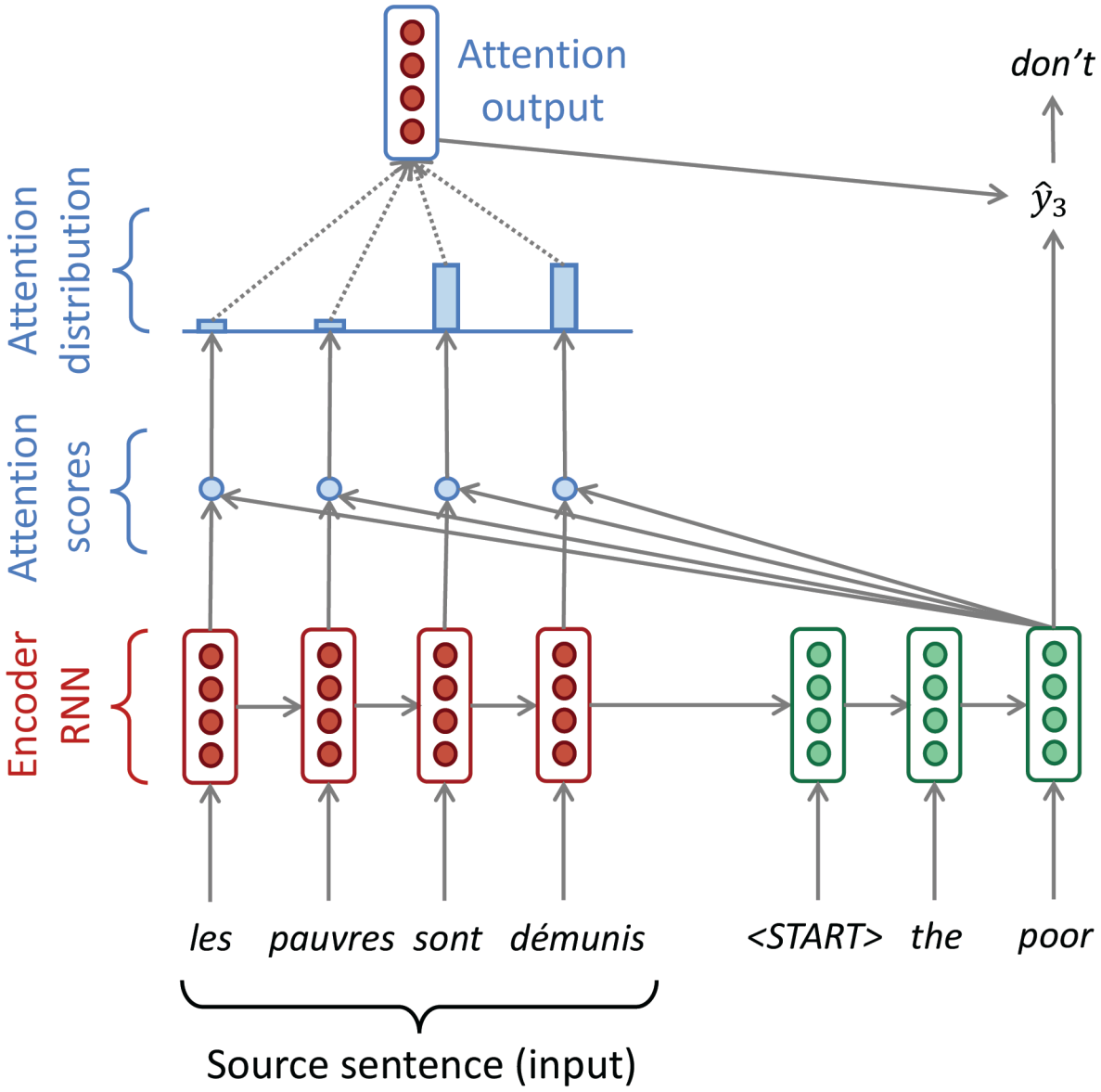
Decoder RNN

== conditional language model

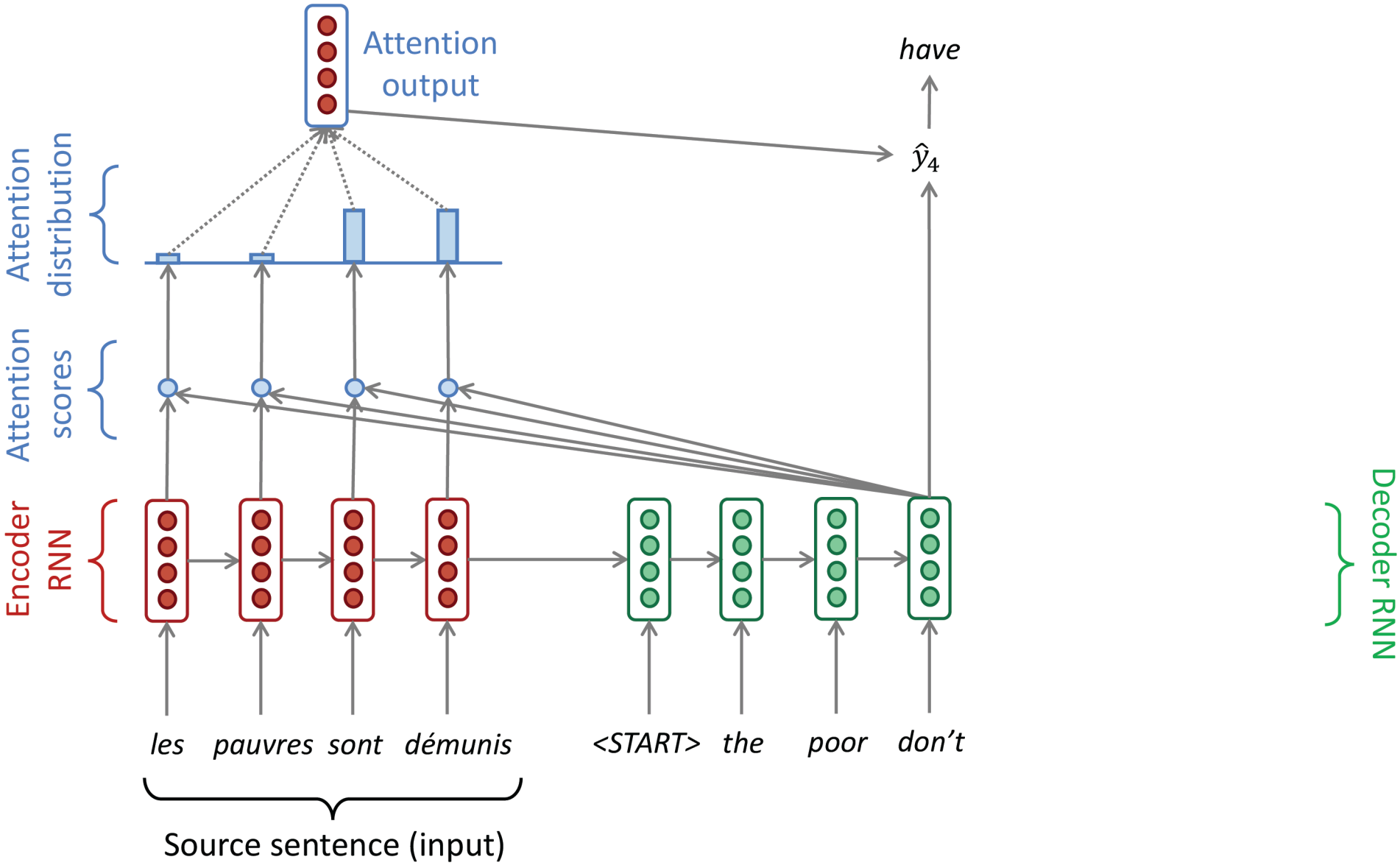
Dot-Product Attention



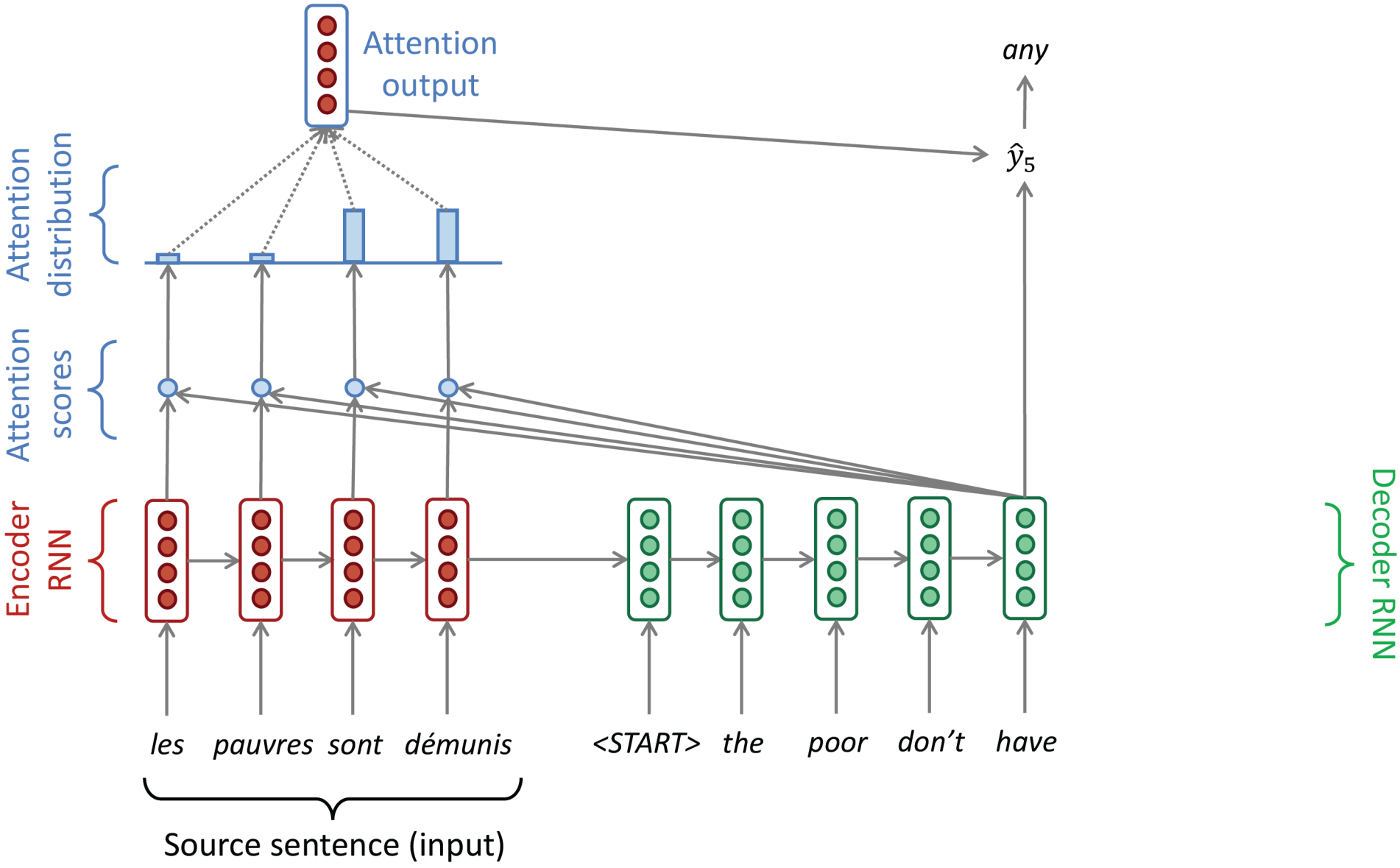
Dot-Product Attention



Dot-Product Attention

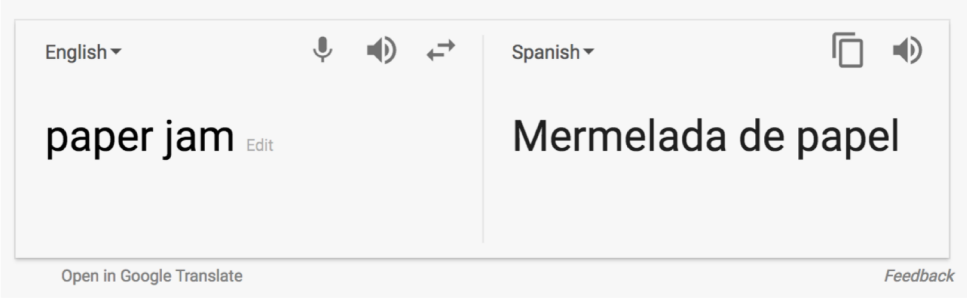


Dot-Product Attention

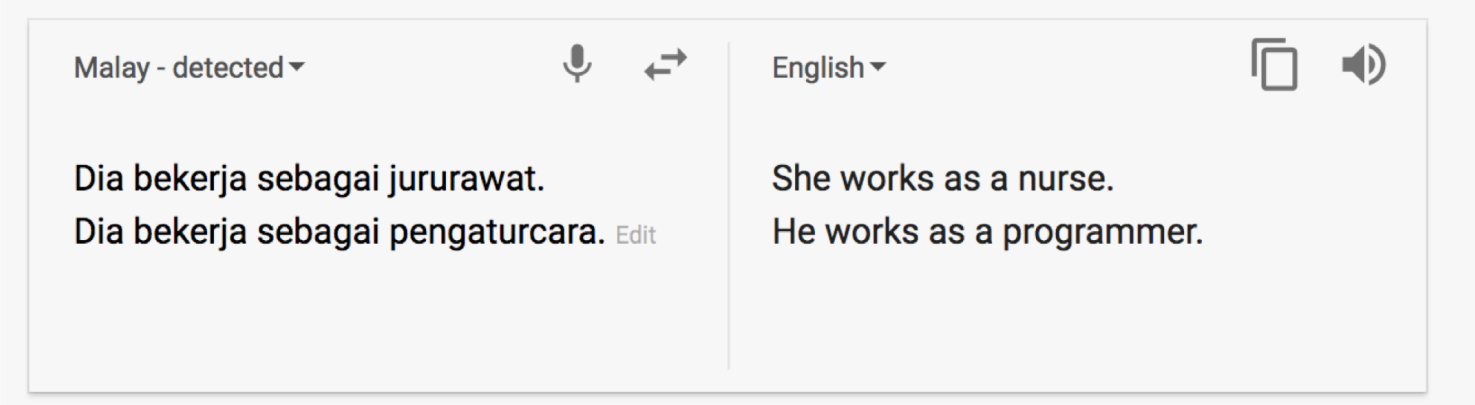


Prevailing Problems [9]

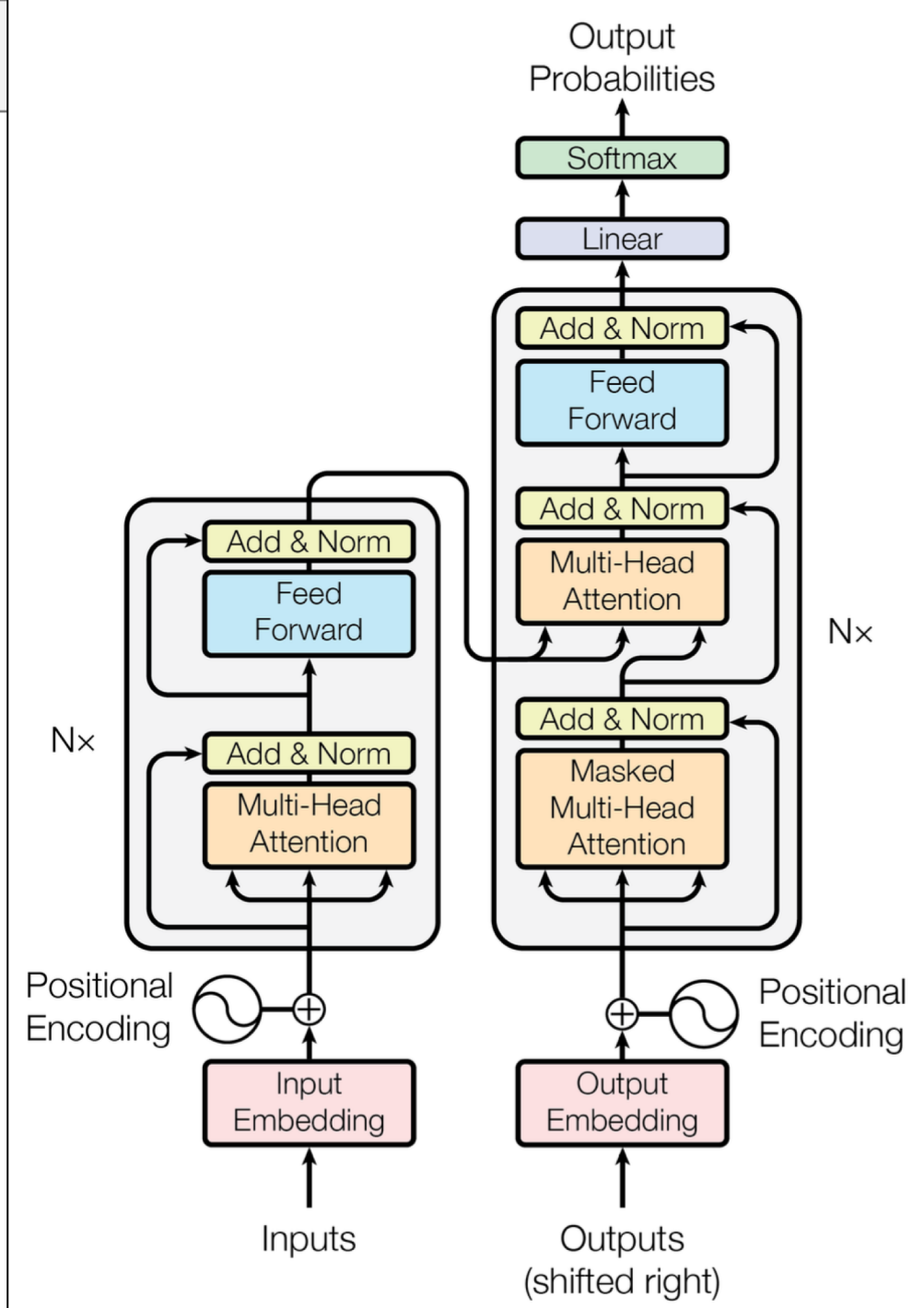
- maintaining context over longer text
- using common sense still hard:



- picks up cultural bias in training data:

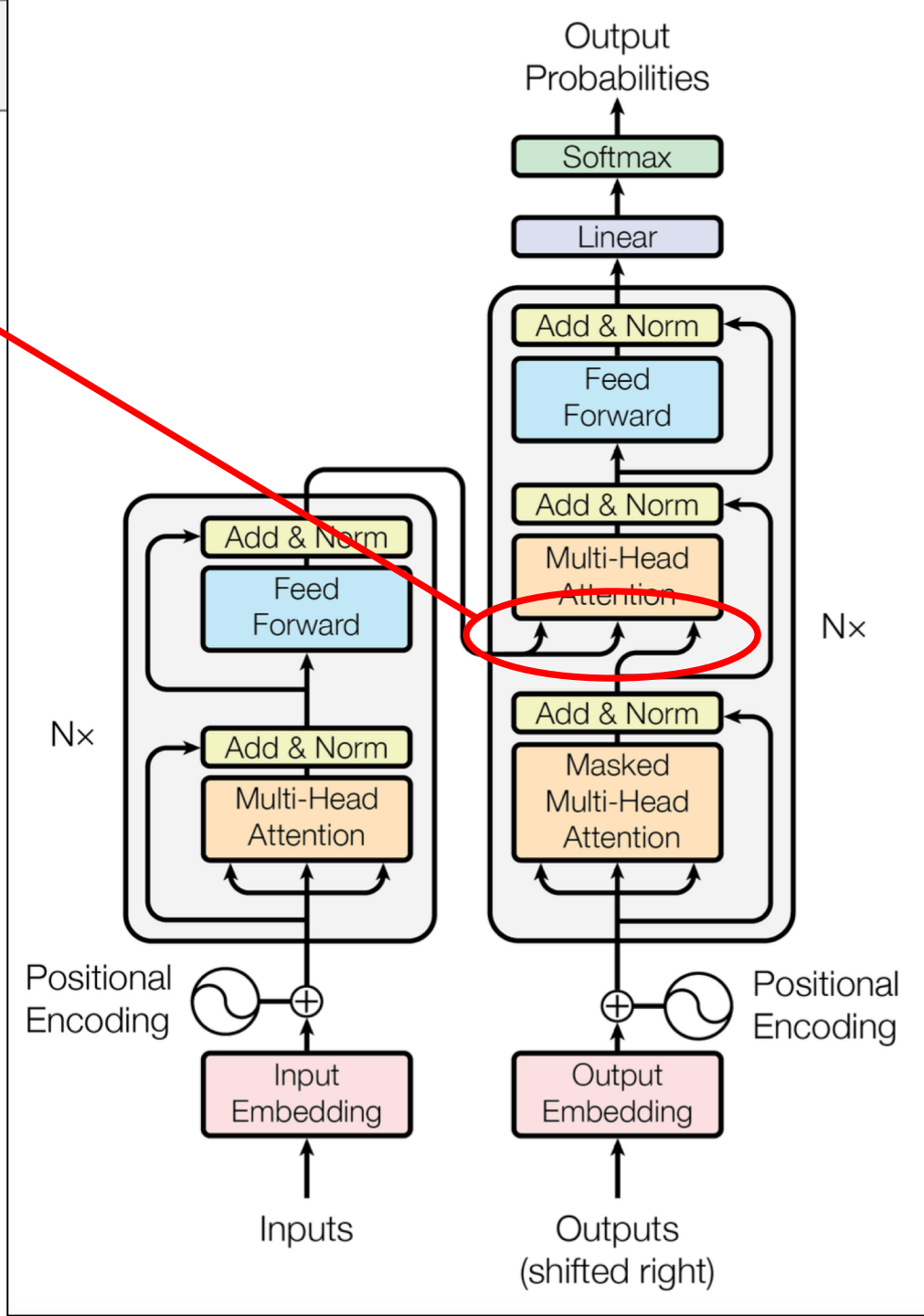


Go Massive: Transformer [1]



Go Massive: Transformer

encoder decoder (cross-)attention :
queries from previous decoder layer;
keys and values from encoder output

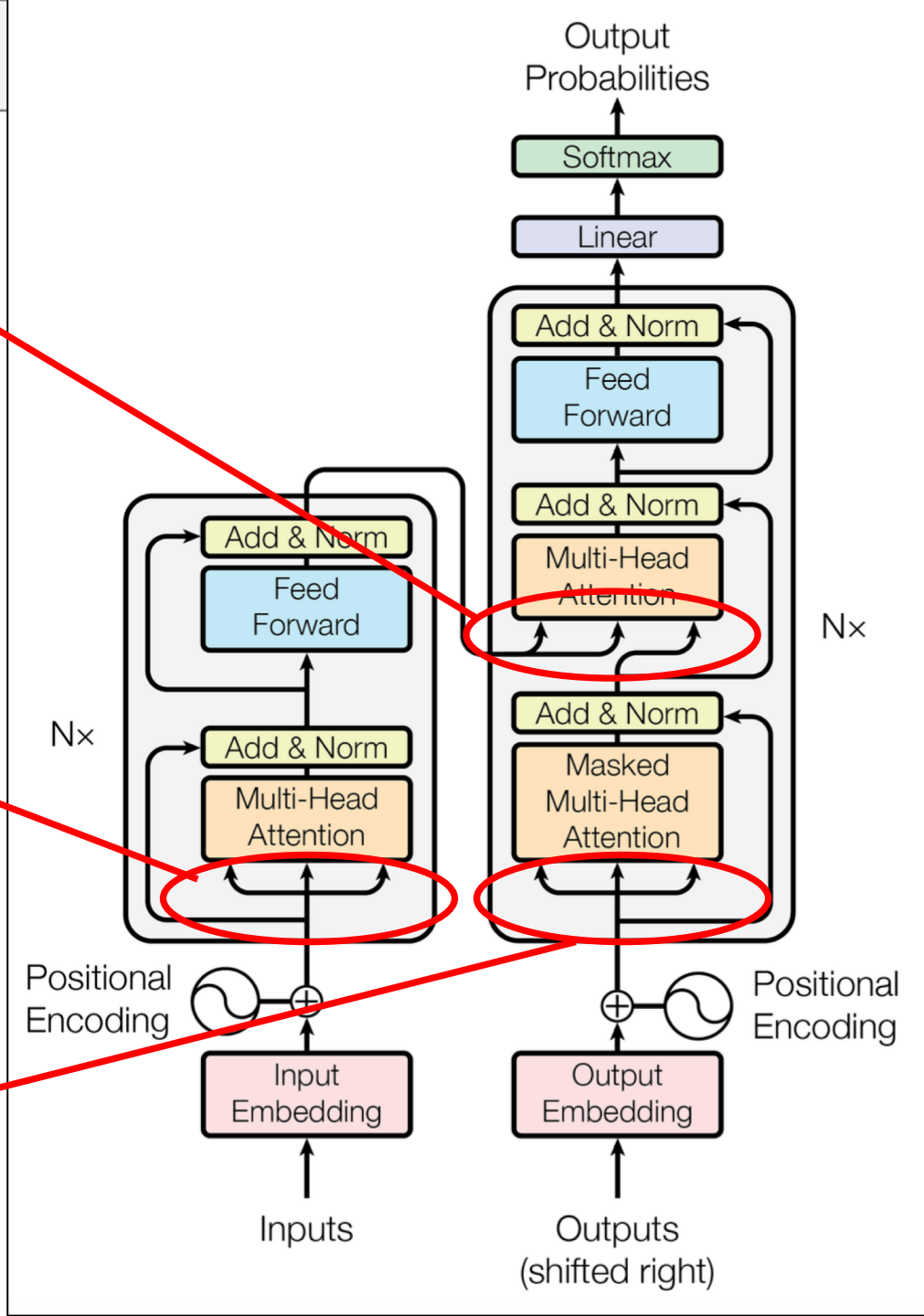


Go Massive: Transformer

encoder decoder (cross-)attention :
queries from previous decoder layer;
keys and values from encoder output

encoder self-attention :
queries, keys, and values from previous encoder layer

(masked) decoder self-attention :
queries, keys, and values from previous decoder layer



BERT [2]

- "BERT": Bidirectional Encoder Representations from Transformers.": a **transformer encoder**

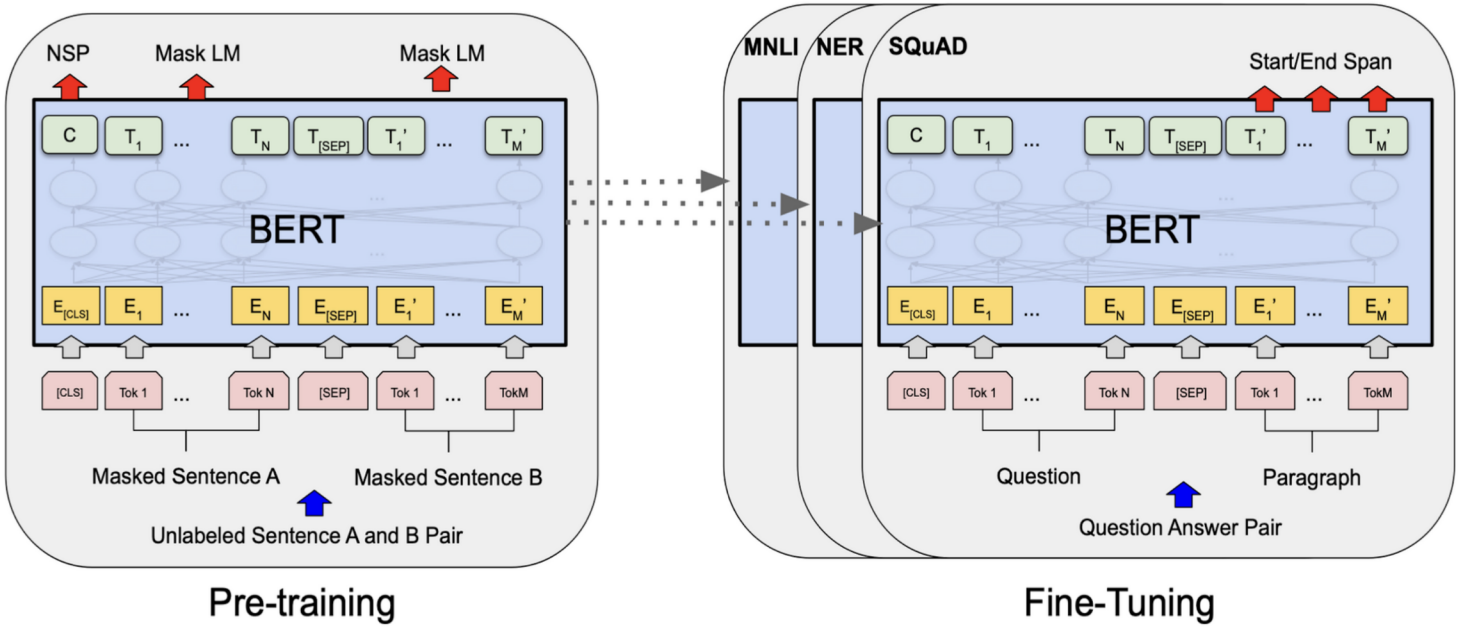
- Trained on BooksCorpus (800 million words) and English Wikipedia (2,500 million words)

- Pre-training tasks
  - Masked word prediction
  - next sentence prediction (NSP)

- BERT versions
  - BERT-Base: 12-blocks, 768-dim-vectors, 12 attention heads. (110M parameters)
  - BERT-Large: 24-blocks, 1024-dim-vectors, 16 attention heads. (340M parameters)

BERT | Use Cases and Extensions

- "Pre-train once, finetune many times"

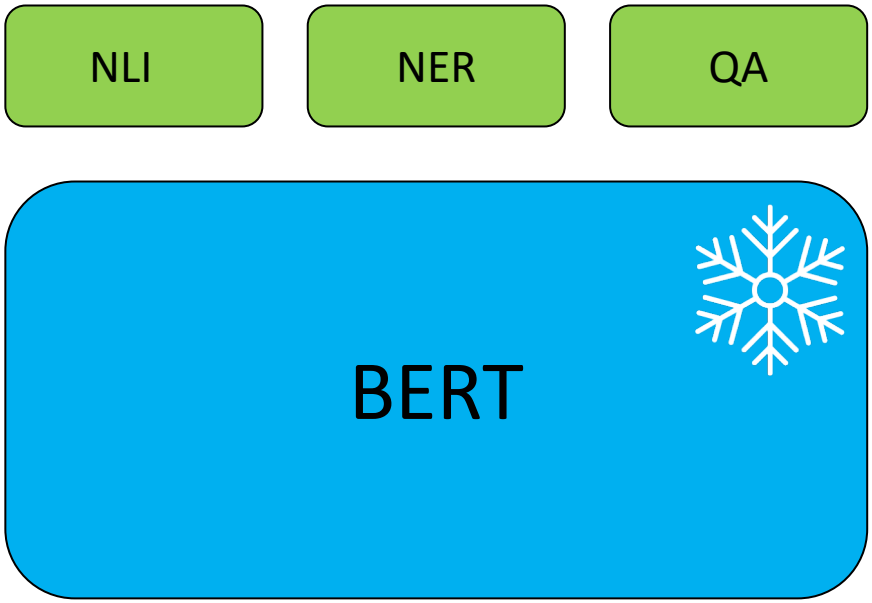


Extensions

- RoBERTa: mainly just train BERT on larger data batches and remove next sentence prediction.
- SpanBERT: masking contiguous spans of words makes a harder, more useful pre-training task
- DistilBERT: a smaller (40%), faster (60%) BERT. Same architecture distilled with a teacher-student setting. It retains 97% of the

BERT | Use Cases and Extensions

- "Pre-train once, finetune many times"



→ need additional parameters for each downstream tasks, e.g. sentiment analysis, MT, NLI, etc.

GPT-3 | Comparison against BERT

	BERT	GPT-3 Generative Pretrained Transformer
Size	340M parameters, trained on ~3.3 Billion tokens	175B parameters, trained on ~500 Billion tokens
Architecture	Bidirectional, made of transformer **encoder** blocks	Autoregressive, made of transformer **decoder** blocks
Training	Masked LM + next sentence prediction	Simple Language Modeling
Usage	Use as contextual encoder + **fine-tune** extra layers on downstream task	Use as-is for any task with **few-shot learning** techniques

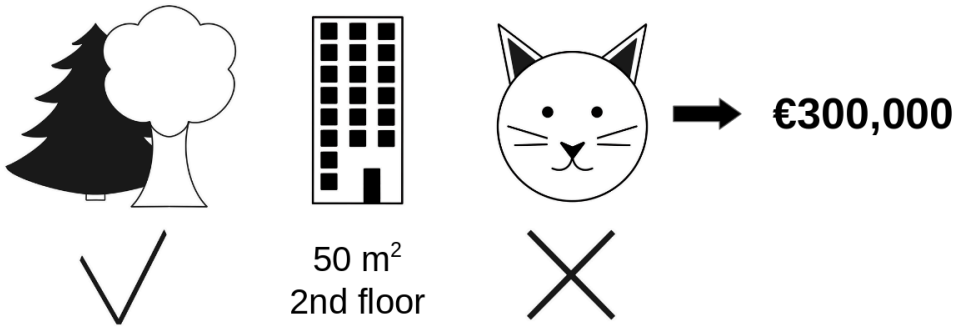
Excursion: SHAP

(see [3] and citations there and [4] (some images taken over from [4]))

To understand the idea of the Shapley values take a look at
https://christophm.github.io/interpretable-ml-book/shapley.html

Shapley Values (see [4])

- Shapley value of a feature-value x_i for a concrete input x on a model predicting $f(x)$: „how important is the contribution of that feature x_i for the prediction?



[4]

Average for all appartments: 310,000

Shapley Values (see [4])

- Shapley value of a feature-value x_i for a concrete input x on a model predicting $f(x)$: „how important is the contribution of that feature x_i for the prediction?

Test coalition of

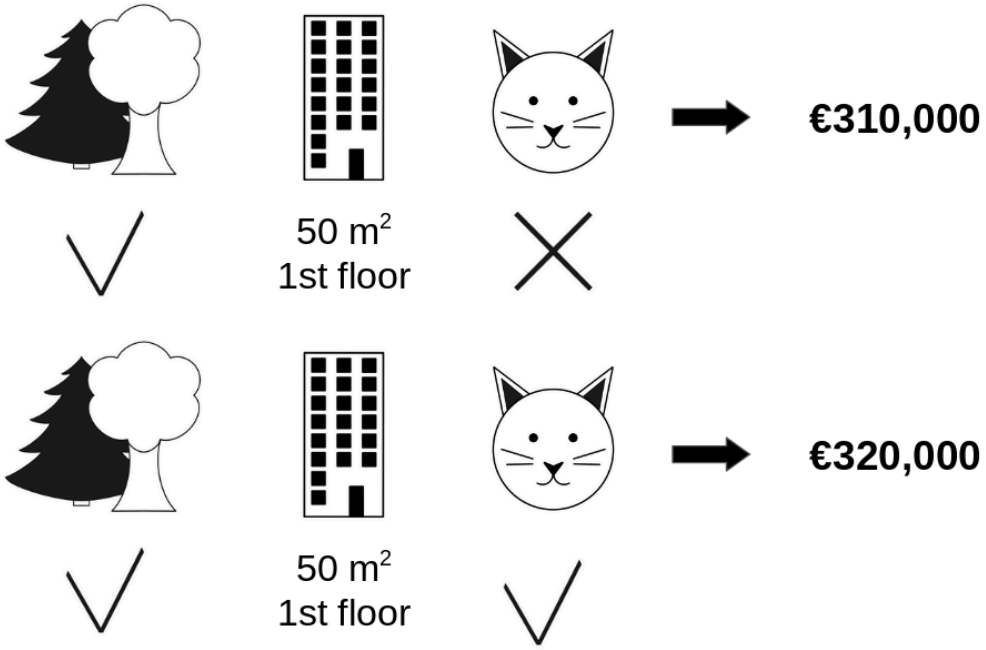
"park-nearby=true" + "sqm=50"

by comparing

"park-nearby=true" + "sqm=50" + "cat-allowed=false" + *random value* for "floor"

against

"park-nearby=true" + "sqm=50" + *random value* for cat-allowed" + *random value* for "floor"



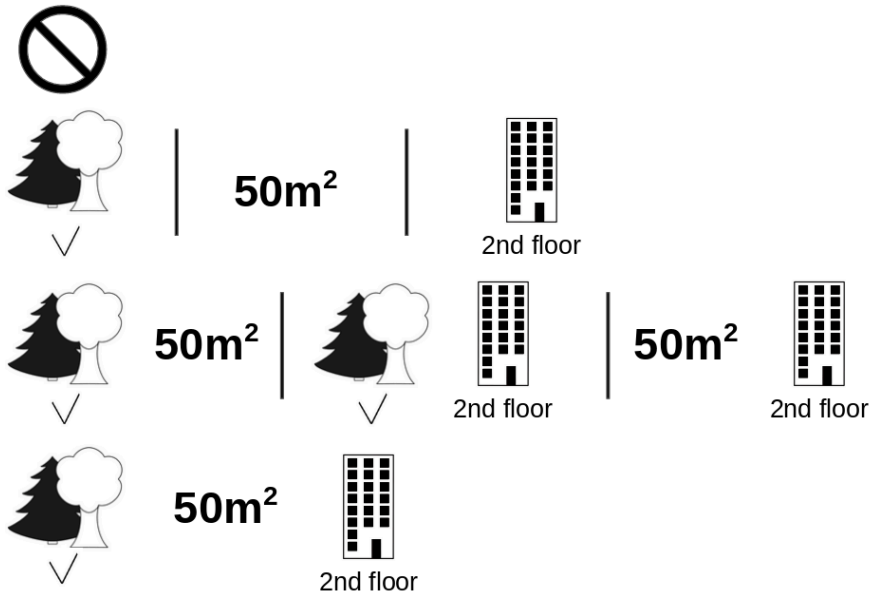
[4]

→ Here: contribution of cat-allowed: -10,000
→ do more random samples for cat-allowed and floor and average

Shapley Values (see [4])

- Shapley value of a feature-value x_i for a concrete input x on a model predicting $f(x)$: „how important is the contribution of that feature x_i for the prediction?

Now test all other 8 possible coalitions = feature combinations against “cat-allowed=false”
and average
(each with a lot of random sampling) and average all → Shapley value of “cat-allowed=false”

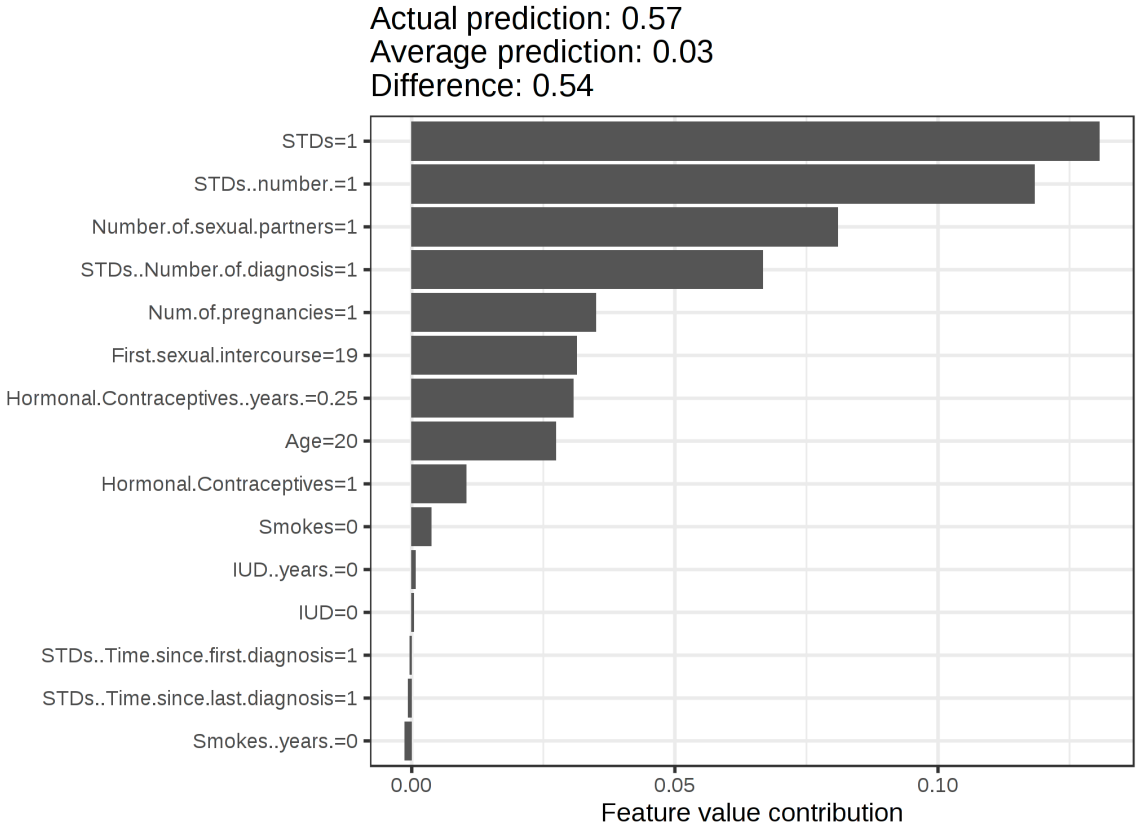


[4]

Shapley Values (see [4])

- Shapley value of a feature-value x_i for a concrete input x on a model predicting $f(x)$: „how important is the contribution of that feature x_i for the prediction?

Case for predicting a class-probability $P(c)$ (here c = cervical cancer) instead of a regression value:



[4]

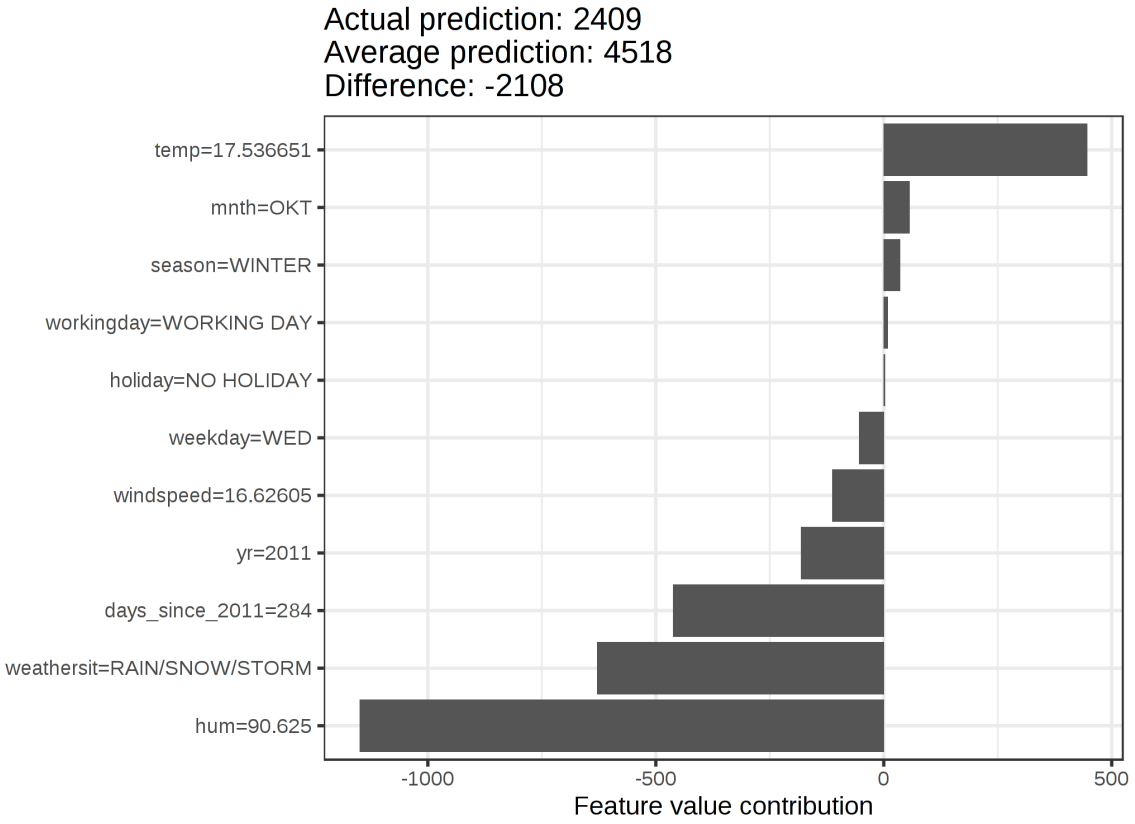
FIGURE 5.46: Shapley values for a woman in the cervical cancer dataset. With a prediction of 0.57, this woman's cancer probability is 0.54 above the average prediction of 0.03. The number of diagnosed STDs increased the probability the most. The sum of contributions yields the difference between actual and average prediction (0.54).

[4]

Shapley Values (see [4])

- Shapley value of a feature-value x_i for a concrete input x on a model predicting $f(x)$: „how important is the contribution of that feature x_i for the prediction?

Case for predicting a regression value (bike rentals)



[4]

FIGURE 5.47: Shapley values for day 285. With a predicted 2409 rental bikes, this day is -2108 below the average prediction of 4518. The weather situation and humidity had the largest negative contributions. The temperature on this day had a positive contribution. The sum of Shapley values yields the difference of actual and average prediction (-2108).

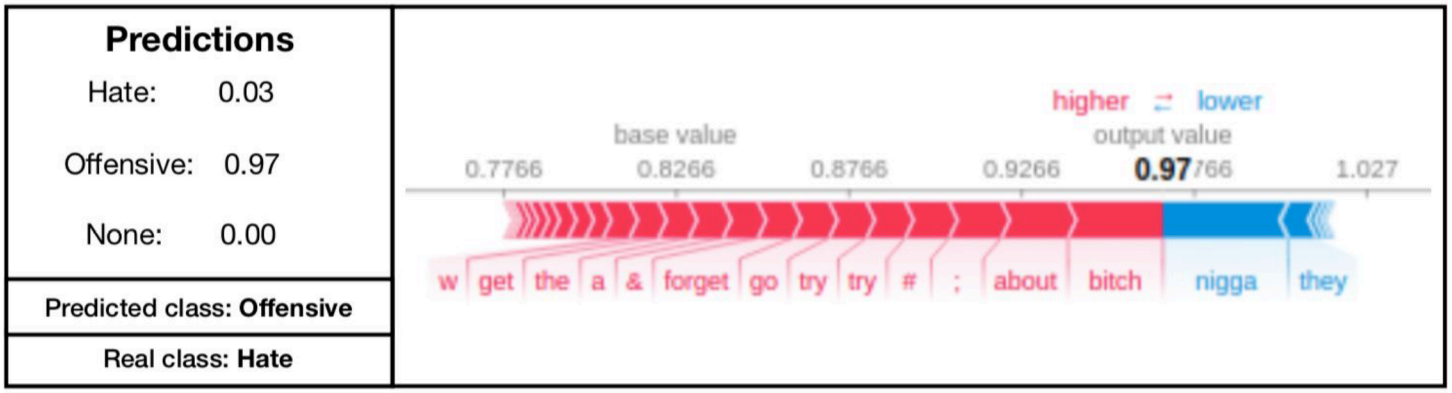
[4]

SHAP Analysis

Consider the example: *RT @s_bitchy: Bitches be fallin so Inlove w. Niggass & then he go get a new bitch & they try go get a new nigga to try to forget about the ….*

- Real class is Hate (1 vote for offensive, 2 for hate, 0 for none)
- BiLSTM predicts Offensive

rt <user>: bitches be fallin so inlove w . niggas & then he go get a new bitch & they try to get a new nigga to try to forget about the & # <number> ;



SHAP Analysis

- Multi Labelling Approach predicts Hate and Offensive

rt <user>: bitches be fallin so inlove w . niggas & then he go get a new bitch & they try to get a new nigga to try to forget about the & # <number> ;

Predictions - Hate	
No:	0.43
Yes:	0.57
Predicted class: Yes	
Real class: Yes	

rt <user>: bitches be fallin so inlove w . niggas & then he go get a new bitch & they try to get a new nigga to try to forget about the & # <number> ;

Predictions - Offensive	
No:	0.01
Yes:	0.99
Predicted class: Yes	
Real class: Yes	

SHAP Analysis

Consider the example: *Ronnie Radke's a fucking fag. Fronz owns his ass.*

- Real class is Offensive (2 votes for offensive, 1 for hate, 0 for none)
- BiLSTM predicts Hate

ronnie radke ‘ s a fucking fag . fronz owns his ass

Predictions	
Hate:	0.56
Offensive:	0.44
None:	0.00
Predicted class: Hate	
Real class: Offensive	

SHAP Analysis

- Multi Labelling Approach predicts Hate and Offensive

ronnie radke ‘ s a fucking fag . fronz owns his ass

Predictions - Hate	
No:	0.18
Yes:	0.82
Predicted class: **Yes**	
Real class: **Yes**	

ronnie radke ‘ s a fucking fag . fronz owns his ass

Predictions - Offensive	
No:	0.01
Yes:	0.99
Predicted class: **Yes**	
Real class: **Yes**	

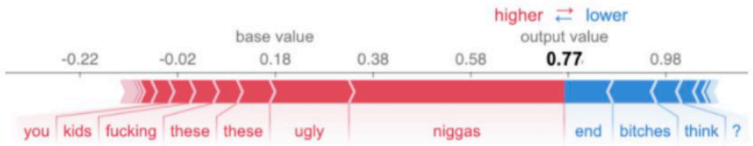
Analysis of Racist Tweets

- 48 tweets annotated as Hate contain the N-word, in the test set of Davidson data.
- The multi-class Bi-LSTM model could correctly classify only 6 tweets out of 48.
- The Multi-Label model could correctly classify 32 out of 48.

why you think all these niggas end up with all these kids ? fucking ugly bitches

Predictions		Comment
Hate: 0.19 Offensive: 0.81 None: 0.00		Single Class Model predicts Offensive
Predicted class: **Offensive**		
Real class: **Hate**		

why you think all these niggas end up with all these kids ? fucking ugly bitches

Predictions		Comment
Not Hate: 0.23 Hate: 0.77		Multi Class model correctly predicts Hate
Predicted class: **Hate**		
Real class: **Hate**		

Submitting Your Solution

As the tutors have to read your Shap explanations:

PLEASE HAND IN YOUR iPython notebook WITH THE OUTPUT!

Submitting Your Solution

- work by expanding the .ipynb iPython notebook for the exercise that you downloaded from Moodle.
- save your expanded .ipynb iPython notebook in your working directory.
  Submit your .ipynb iPython notebook + HTML version via Moodle (nothing else)
- remember: working in groups is not permitted.
  Each student must submit her own ipynb notebook!
- we check for plagiarism. Each detected case will have the consequence of 5.0 for the whole exercise grade.
- deadline: please check Moodle

Citations

[1] Waseem, Z., & Hovy, D. (2016). Hateful symbols or hateful people? Predictive features for hate speech detection on Twitter. In Proceedings of the naacl student research workshop (pp. 88-93).

[2] Georg Groh: Lecture NLP, TUM

[3] Edoardo Mosca et al: Talk at SocialNLP @ NAACL, June 2021

[4] Christoph Molnar: Interpretable Machine Learning - A Guide for Making Black Box Models Explainable. 2021, Online book: https://christophm.github.io/interpretable-ml-book/ (URL, June 2021)

References

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L. and Polosukhin, I., (2017). Attention is all you need. In Advances in Neural Information Processing Systems (pp. 5998-6008) and arXiv:1706.03762v5

[2] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[3] Brown, Tom, et al. "Language models are few-shot learners." *Advances in neural information processing systems* 33 (2020): 1877-1901.

[4] RAFFEL, Colin, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 2020, 21. Jg., Nr. 1, S. 5485-5551.

[5] BORGEAUD, Sebastian, et al. Improving language models by retrieving from trillions of tokens. In: *International conference on machine learning*. PMLR, 2022. S. 2206-2240.

[6] ROMBACH, Robin, et al. High-resolution image synthesis with latent diffusion models. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022. S. 10684-10695.

[7] AMATRIAIN, Xavier. Transformer models: an introduction and catalog. *arXiv preprint arXiv:2302.07730*, 2023.

[8] Kaplan, Jared, et al. "Scaling laws for neural language models." *arXiv preprint arXiv:2001.08361* (2020).

[9] Richard Socher et al: “CS224n: Natural Language Processing with Deep Learning”, Lecture Materials http://web.stanford.edu/class/cs224n/ (URL, May 2018), 2018

References

[10] Xiao, H. (2023) : Auto-GPT Unmasked: The Hype and Hard Truths of Its Production Pitfalls https://jina.ai/news/auto-gpt-unmasked-hype-hard-truths-production-pitfalls/ (URL, May 2023

[11] Belatgy et al (2022): Zero- and Few-Shot NLP with Pretrained Language Models - ACL Tutorial 2022

[12] Schick, T., Schütze, H. (2020) : Exploiting Cloze Questions for Few Shot Text Classification and Natural Language Inference https://arxiv.org/abs/2001.07676

[13] Liu et al (2022): Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning https://arxiv.org/abs/2205.05638

[14] Sanh V. (2022): Multitask Prompted Training Enables Zero-Shot Task Generalization https://arxiv.org/abs/2110.08207