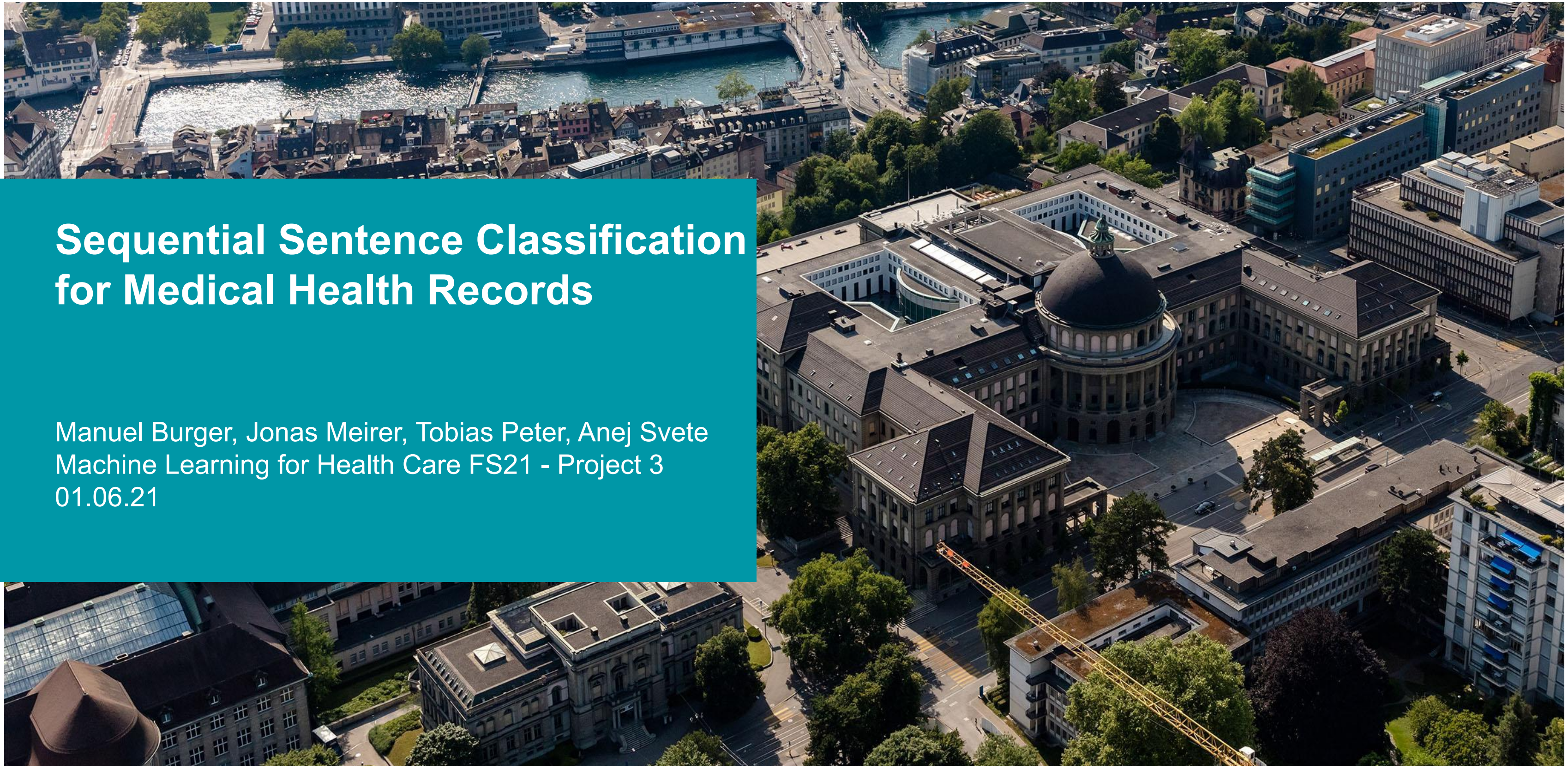
ETH zürich

Sequential Sentence Classification for Medical Health Records

Manuel Burger, Jonas Meirer, Tobias Peter, Anej Svete
Machine Learning for Health Care FS21 - Project 3
01.06.21

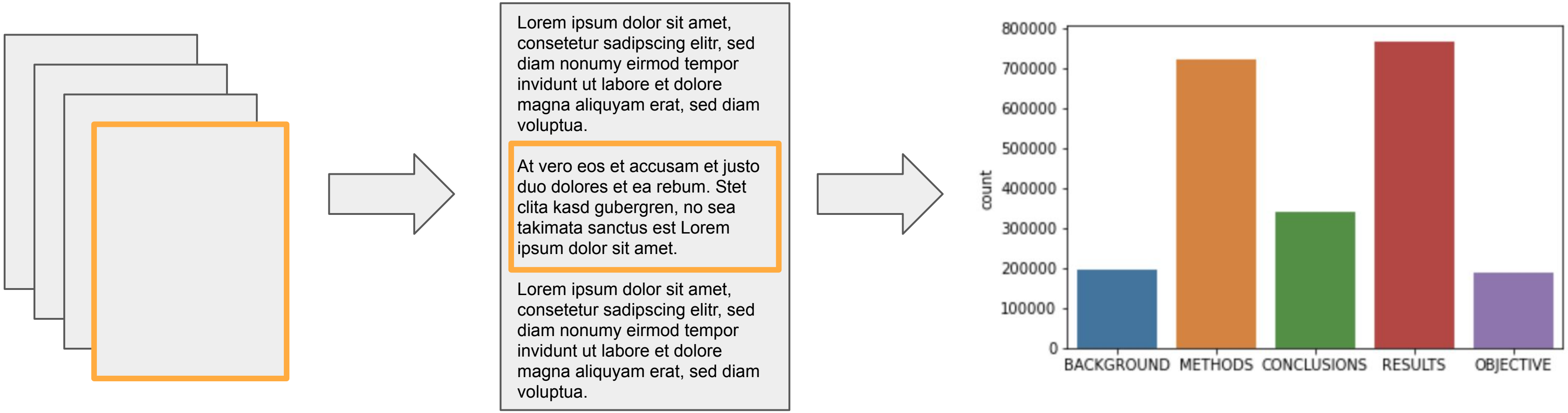


Outline

- Task Overview
- Word Embeddings (Word2Vec, FastText)
- Keras Text Vectorization and LSTM
- Transfomers
- Hierarchical modelling with Transformer and RNN
- Conclusion

Task Overview

- Sentence classification on PubMed 200k RCT dataset



Word Embeddings

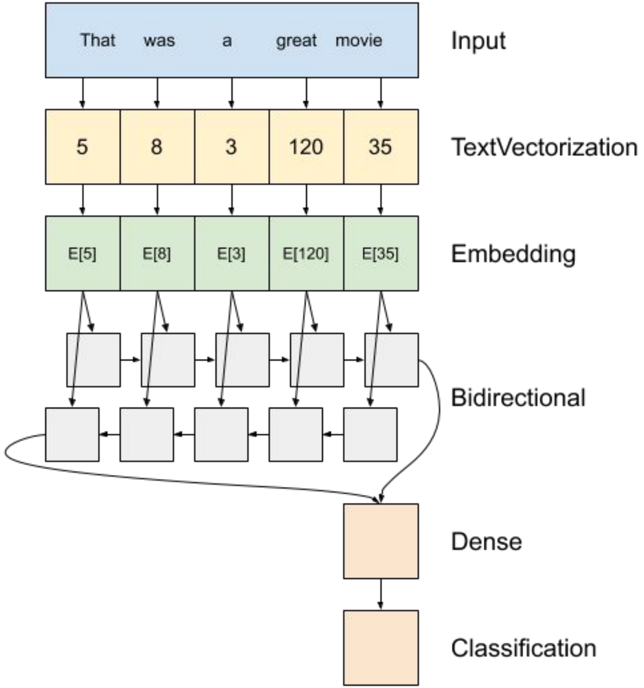
1. Trained Word2Vec as well as fastText.
2. Hyperparameter search (e.g. dimension, min. count) in combination with LogReg & XGBoost Classifier
   → embedding dimension of 250/300
3. W2V & fastText yielded similar performance:
   - With LogReg both slightly worse than Tf-idf
   - XGBoost yields only additional 3 pp over LogReg
4. fastText's *supervised* algorithm (indeed a very fast off-the-shelf baseline) with 0.85 F1-score.
5. MLP: concatenated W2V mean embeddings of neighboring sentences (3 on each side) of the same abstract for additional context:
   - highest F1-score (0.87) with mean embeddings.

Results with Mean Embeddings

F1-score	LogReg	XGB
W2V	0.76	0.79
fastText	0.75	0.78
Tf-idf	0.77	-

fastText supervised	0.85
MLP	**0.87**

Keras Text Vectorization and LSTM



Text Vectorization

```
vectorize_layer = TextVectorization(max_tokens=VOCAB_SIZE,
                                    output_mode='int',
                                    ngrams=1,
                                    output_sequence_length=MAX_SEQUENCE_LENGTH)
vectorize_layer.adapt(X_train)
```

Model Setup

```
model = Sequential()

model.add(Input(shape=(1,), dtype=tf.string))

model.add(vectorize_layer)

model.add(Embedding(VOCAB_SIZE, EMBEDDING_SIZE, trainable=True))

model.add(Bidirectional(LSTM(EMBEDDING_SIZE,
                             dropout=0.4,
                             recurrent_dropout=0.25,
                             return_sequences=True)))

model.add(Bidirectional(LSTM(EMBEDDING_SIZE,
                             dropout=0.4,
                             recurrent_dropout=0.25)))

model.add(BatchNormalization())
model.add(Dense(256, activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(0.3))
model.add(Dense(256, activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(0.3))

model.add(Dense(256, activation='relu'))
model.add(Dense(N_CLASSES, activation='softmax'))
```

Notes:

- Does not work well with preprocessed sentences
- Performance Boost (micro-F1) from 87% to 90%

Transformers

- **Transformers** are current state-of-the-art models across various tasks
  - **BERT-based** models known to perform best for tasks like classification
- The large models are **pre-trained on enormous corpora** and then **fine-tuned for any task of interest**
  - In our case, that was sentence classification
- The **Huggingface () library** provides numerous implementations and pre-trained models
- They can **take into account the word/sentence context** and provide a **richer representation than raw word embeddings**

Raw sentence

Based on the findings, we conclude...

Model-dependent tokenization

[1, 2007, 312, 0, 281, 2987, 482, ...]

magic

(109 million parameters)

[0.821, 0.127, 1.928, 0.216, 2.167, ...]

A Classifier head

Prediction

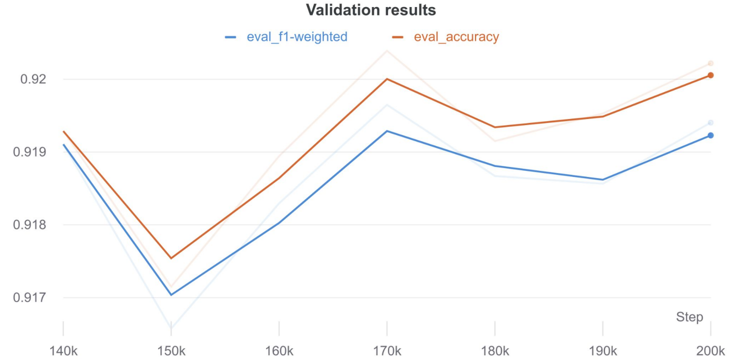
Transformers

Base model

- We used a pure BERT model **pre-trained on a corpus of full PubMed texts [3]** and added a **classifier head**
  - It "knows" the vocabulary well
  - It performed better than a model trained on unrelated text

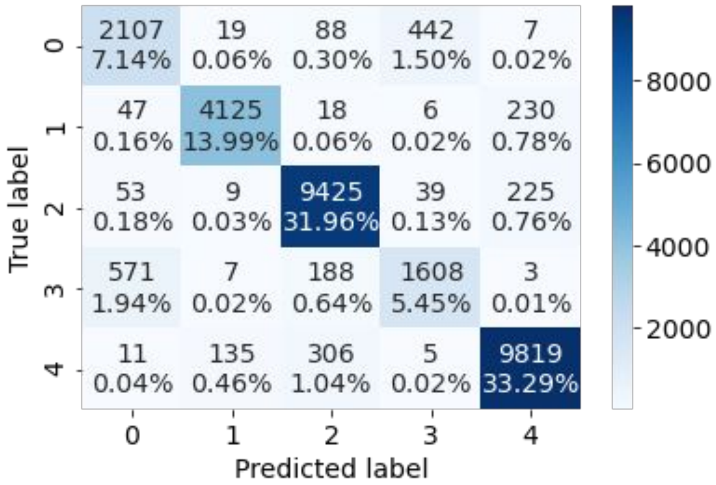
Fine-tuning

- The default tokenizer was used on sentences with context window of size 3
  - Sequence length was capped at 96
  - The special way the library performs tokenization is beneficial
- The **whole model** was fine tuned for **2 epochs** on our training data
  - That takes around 12 hours!
  - Small step size (5e-5), weight decay (0.01) on the whole model



Test set performance

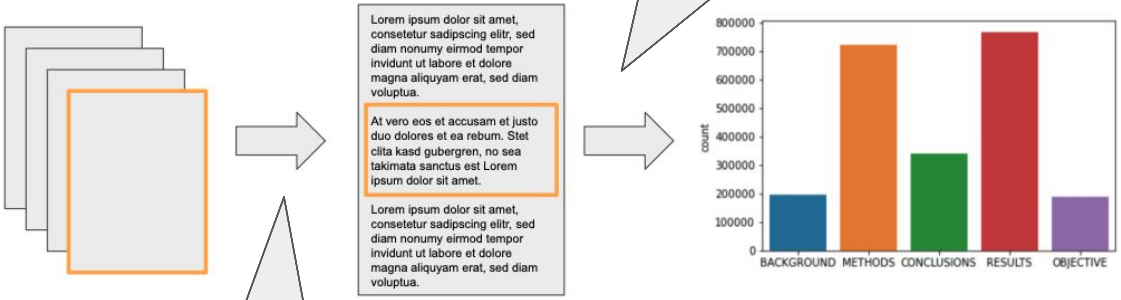
Model	F1 micro	F1 weighted
Plain BERT model	0.918	0.918



ETH zürich

Hierarchical modelling

Classification per sentence



Sentences follow a global structure within an abstract

Inspired by Jin et al. [1] **"Hierarchical Neural Networks for Sequential Sentence Classification in Medical Scientific Abstracts"**

Hierarchical modelling

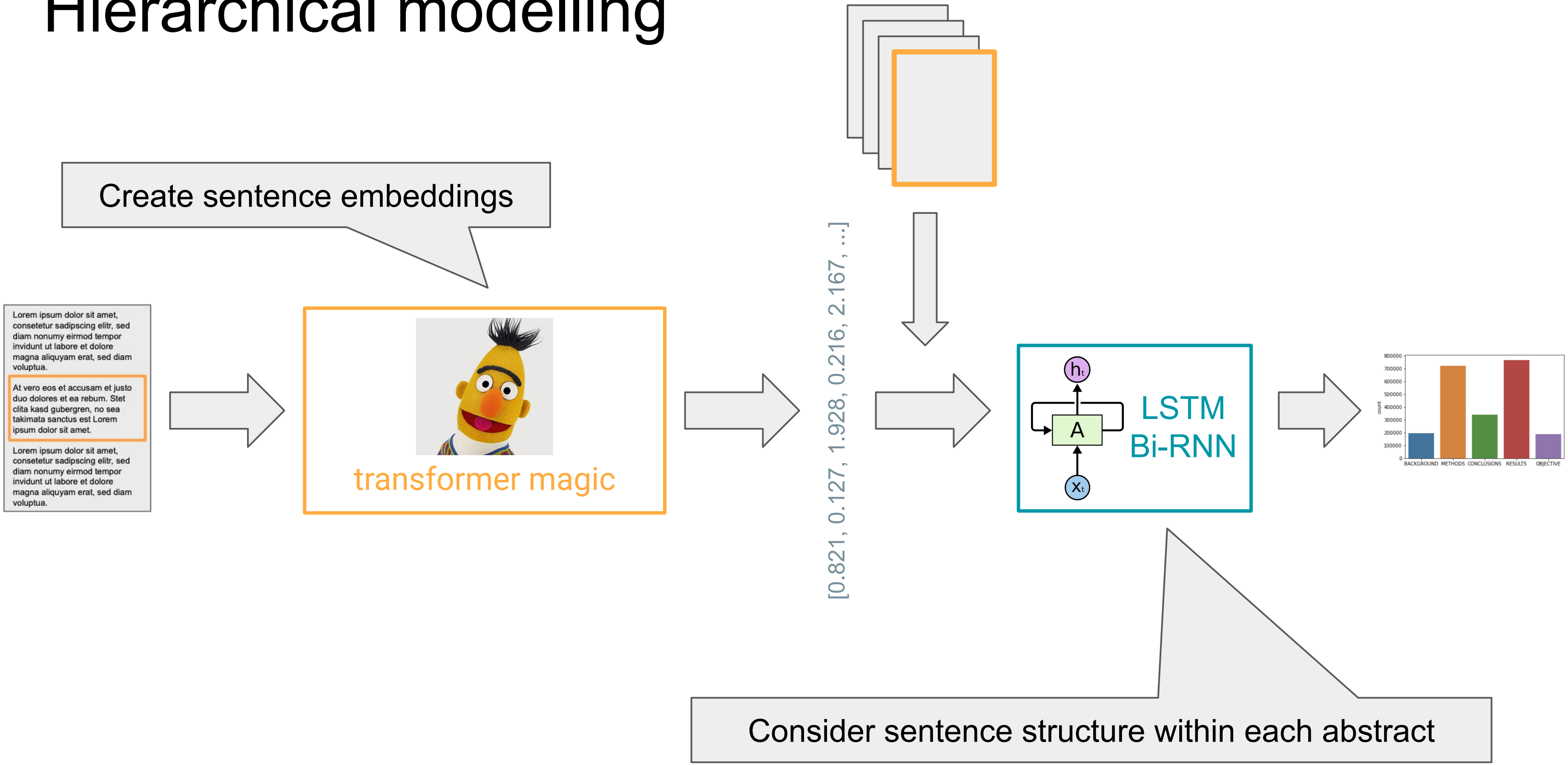
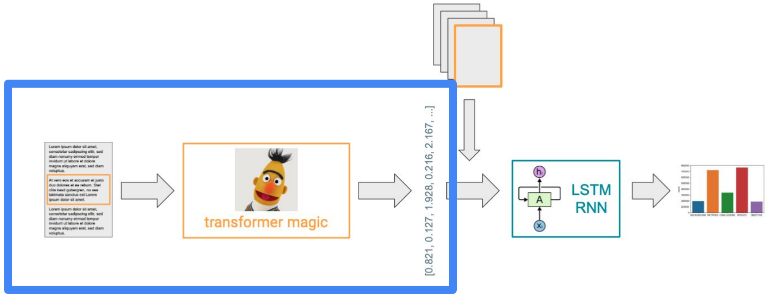


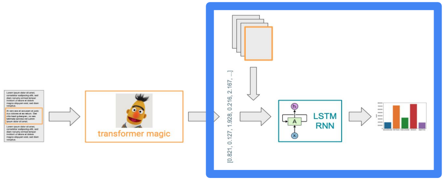
Image Source RNN: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

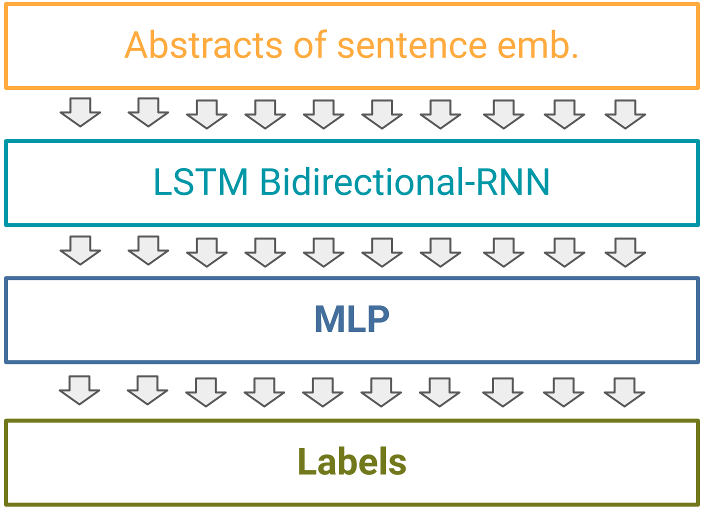
Hierarchical modelling



- BERT language model
- Pretrained on PubMed text data
- Fine-tune complete model to task
  - Attach dense head
  - Train for single epoch
- Predict sentence embeddings for full dataset
  - Averaged token embedding

Hierarchical modelling





- For each abstract:
  - Create sequence of sentence embeddings
  - 0-pad to longest abstract (embedding and label)
- Feed dataset of abstracts through RNN
- Feed each RNN hidden state through an MLP
  - Predict seq. of final class label on hidden state
  - Post-Process: cut-back seq., impute labels

F1 Score: 0.94366 (SOTA)

Conclusion

- Incremental performance improvement through models
- Exploit problem structure and create respective inductive bias

- Improvements:
  - End-to-end training of hierarchical approach
  - Jin et al.:
    - Labelling noise
    - Unsupervised data

Model	F1-Score
Tf-idf (LR)	0.77
W2V (XGB)	0.79
fastText (XGB)	0.78
fastText (supervised)	0.85
MLP	0.87
RNN	0.90
Transformer	0.92
Hierarchical Model	0.944
Jin et al.	0.939

ETH zürich

Thank you for your attention!

Manuel Burger, Jonas Meirer, Tobias Peter, Anej Svete
Machine Learning for Health Care FS21 - Project 3
01.06.21

References

[1] Jin, Di and Szolovits, Peter. **Hierarchical Neural Networks for Sequential Sentence Classification in Medical Scientific Abstracts** (2018). https://www.aclweb.org/anthology/D18-1349

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. **BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding** (2019). https://www.aclweb.org/anthology/N19-1423

[3] Liu, Nigel. **Self-Alignment Pretraining for Biomedical Entity Representations**. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 4228–4238). Association for Computational Linguistics, 2021.