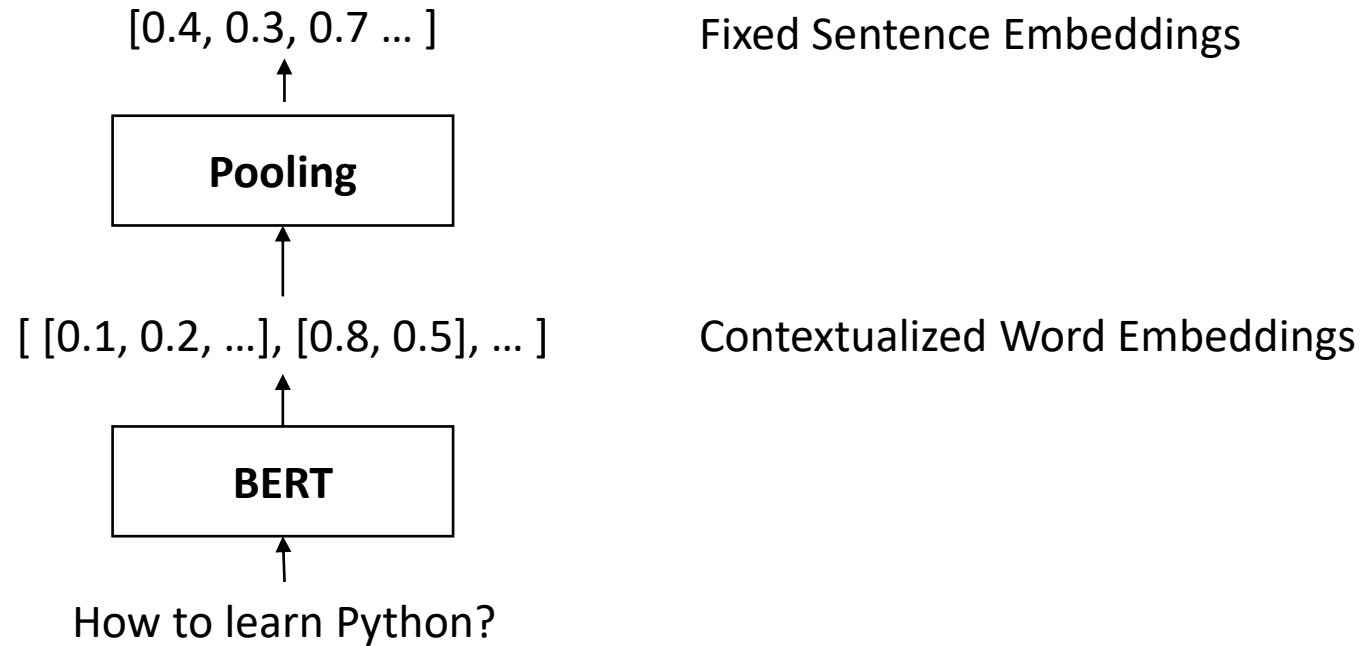


Training of State-of-the-Art Sentence Embedding Models

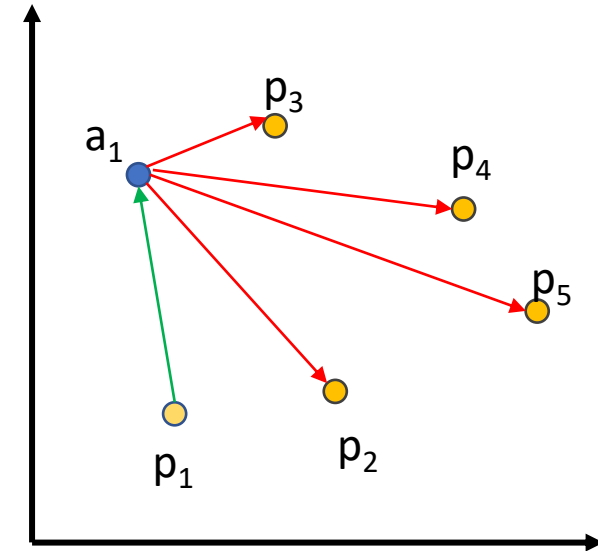
Nils Reimers
Hugging Face
www.SBERT.net

Sentence Embeddings Model



Multiple Negative Ranking Loss

- Have positive pairs:
 - (a_1, p_1)
 - (a_2, p_2)
 - (a_3, p_3)
- Examples:
 - (query, answer-passage)
 - (question, duplicate_question)
 - (paper title, cited paper title)
- (a_i, p_i) should be close in vector space
- (a_i, p_j) should be distant in vector space ($i \neq j$)
 - Unlikely that e.g. two randomly selected questions are similar
- Also called “training with in-batch negatives”, InfoNCE or NTXentLoss



Multiple Negative Ranking Loss

- Mathematical Definition

$$L = -\frac{1}{n} \sum_{i=1}^n \frac{\exp(\text{sim}(a_i, p_i))}{\sum_j \exp(\text{sim}(a_i, p_j))}$$

- Sim: Similarity function between (a, p)
 - Cosine-Similarity
 - Dot-Product

Multiple Negative Ranking Loss

Intuitive Explanation

- a_1 : How many people live in Berlin?
 - p_1 : Around 3.5 million people live in Berlin
 - p_2 : Washington DC is the capital of the US
 - p_3 : The 2021 Olympics are held in Japan
- Compute text embeddings & compute similarities:
 - $\text{sim}(a_1, p_1) = 0.5$
 - $\text{sim}(a_1, p_2) = 0.3$
 - $\text{sim}(a_1, p_3) = 0.1$
- See it as classification task and use Cross-Entropy Loss:
 - Prediction: $[0.5, 0.3, 0.1]$
 - Gold: $[1, 0, 0]$

Multiple Negative Ranking Loss

Intuitive Explanation

- (a_1 : How many people live in Berlin?, p_1 : Around 3.5 million people live in Berlin)
(a_2 : What is the capital of the US?, p_2 : Washington DC is the capital of the US)
(a_3 : Where are the Olympics this year?, p_3 : The 2021 Olympics are held in Japan)

- Compute text embeddings & compute similarities:

$$\text{sim}(\text{vec_a}, \text{vec_b}) = \text{vec_a} * \text{vec_b}^T = \begin{bmatrix} \text{sim}(a_1, p_1), \text{sim}(a_1, p_2), \text{sim}(a_1, p_3) \\ \text{sim}(a_2, p_1), \text{sim}(a_2, p_2), \text{sim}(a_2, p_3) \\ \text{sim}(a_3, p_1), \text{sim}(a_3, p_2), \text{sim}(a_3, p_3) \end{bmatrix}$$

- See it as classification task and use Cross-Entropy Loss:

- Gold: $\begin{bmatrix} 1, & 0, & 0, \\ 0, & 1, & 0, \\ 0, & 0, & 1 \end{bmatrix}$

Multiple Negatives Ranking Loss Code

```
scores = self.similarity_fct(embeddings_a, embeddings_b) * self.scale
labels = torch.tensor(range(len(scores)), dtype=torch.long, device=scores.device) # Example a[i] should match with b[i]
return self.cross_entropy_loss(scores, labels)
```

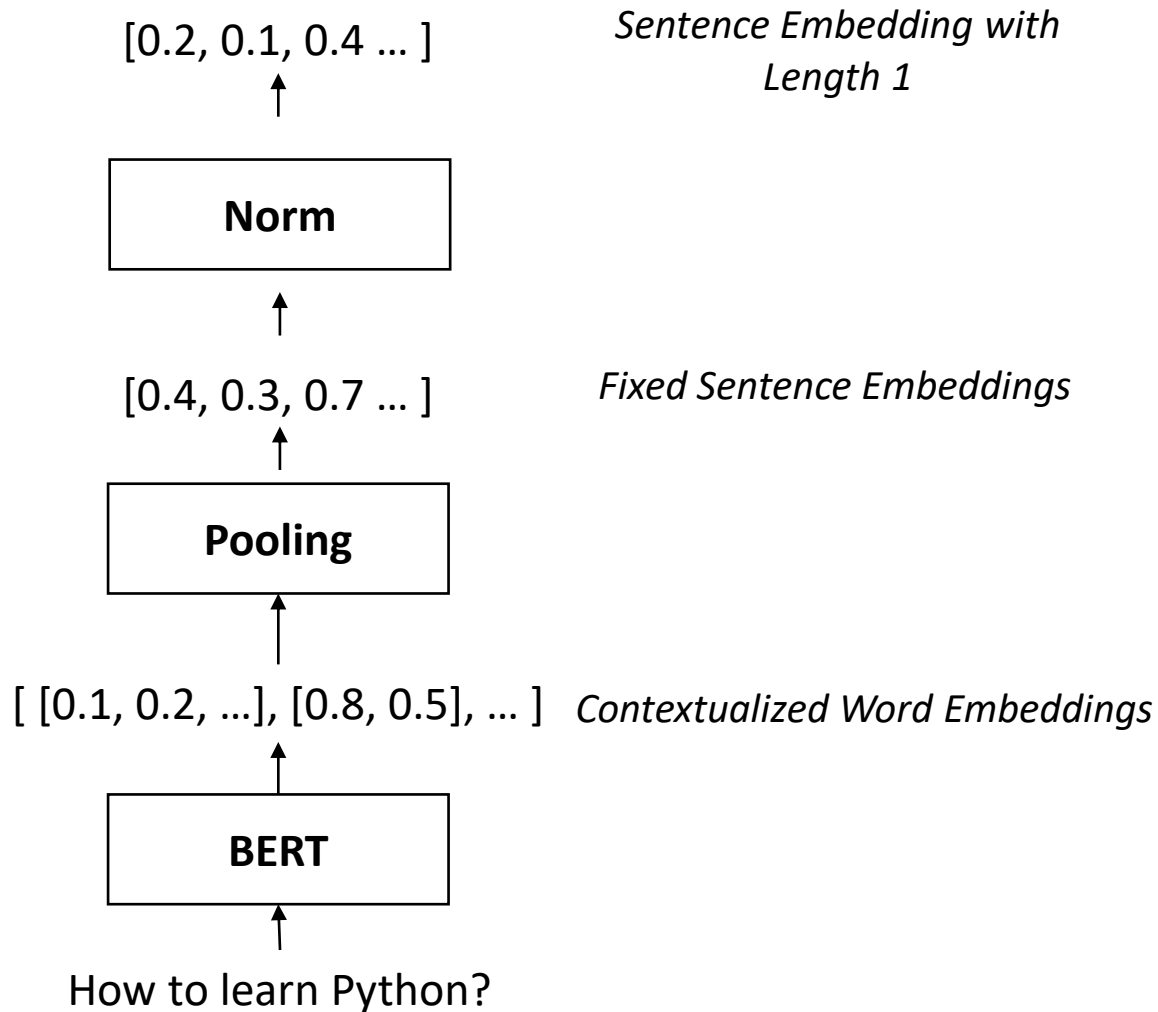
https://github.com/UKPLab/sentence-transformers/blob/master/sentence_transformers/losses/MultipleNegativesRankingLoss.py

Multiple Negatives Ranking Loss

Similarity Functions

- How to compute $\text{sim}(a, b)$?
 - a, b are vectors
 - Dot-product: $\text{dot_prod}(a, b) = ab^T$
 - Cosine-Similarity: $\text{cos_sim}(a, b) = (ab^T) / (||a|| ||b||)$
 - Does not work well, scores differences are too small
 - Scaled Cosine-Similarity: $\text{scaled_cos_sim}(a, b) = C * \text{cos_sim}(a, b)$
 - Works well with e.g. $C=20$
 - Scaled dot-product: $\text{scaled_dot_prod}(a, b) = C * \text{dot_prod}(a, b)$

Network Architecture – For Training with Cosine Similarity



- Cosine-Similarity becomes simply dot-product
- Euclidean-distance is proportional to cosine-similarity
 - K-means clustering is now based on cossim
- Must use `scaled_dot_prod` for training

Cosine-Similarity vs. Dot-Product

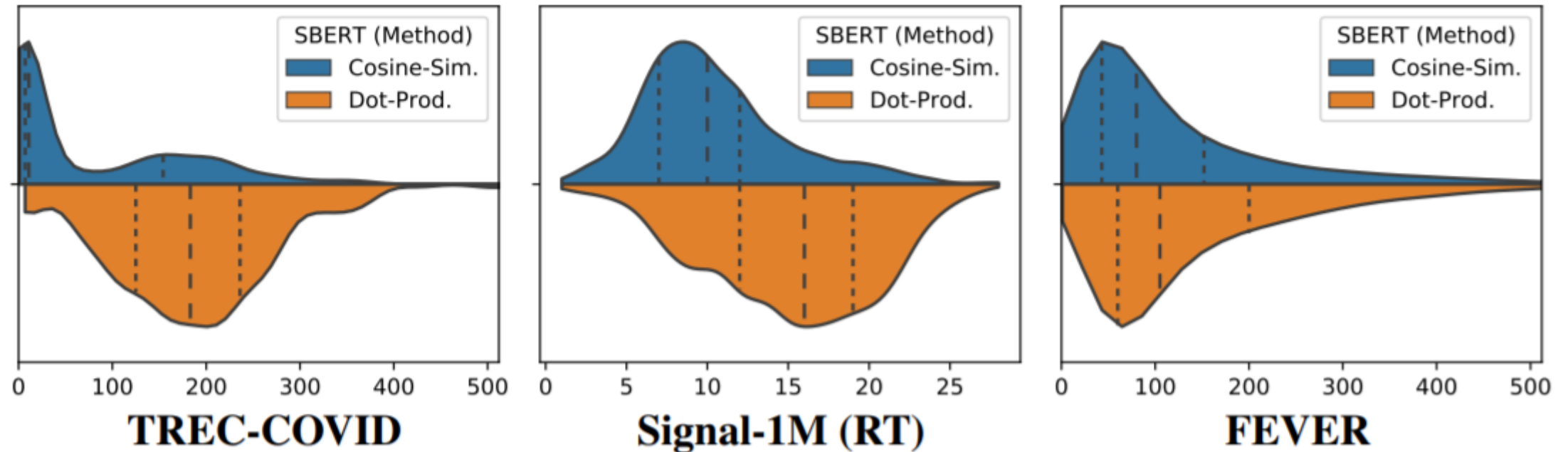
Cosine-Similarity

- Vector has highest similarity to itself
 - $\text{cos_sim}(a, a) = 1$
- With normalized vectors, equal to dot_product
 - With max vector length = 1
- With normalized vectors, proportional to Euclidian distance
 - Works with k-means clustering

Dot-Product

- Other vectors can have higher dot-product
 - $\text{dot}(a, a) < \text{dot}(a, b)$
- Might be slower with certain approximate nearest neighbor methods
 - Max vector length not know
- Does not work with k-means clust.

Cosine-Similarity vs. Dot-Product



- Semantic search: Given short query, find longer passage
- Cosine-Similarity: Prefers retrieval of short passages close to query
- Dot-Product: Prefers longer passages (longer passage = longer vector = higher dot product)

Optimizing the Multiple-Negatives-Ranking-Loss

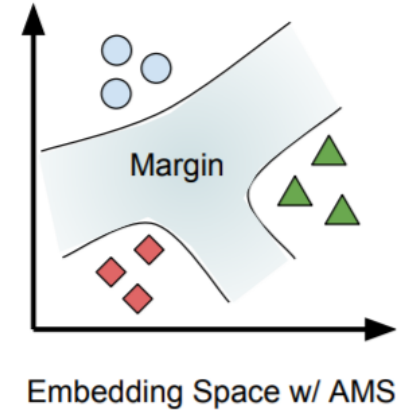
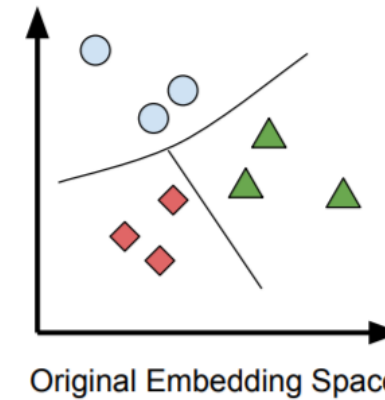
- Training with $\text{scaled_cos_sim}(a, b) = C * \text{cos_sim}(a, b)$
 - How to choose the scale C ? \leq unclear, common values 14-20
 - ConveRT paper: Start at 1, end at 23, increase over first 10k steps
 - CLIP paper: $\text{scaled_cos_sim}(a, b) = \exp(C) * \text{cos_sim}(a, b)$ with C a learnable parameter
 - Unclear impact
 - Will it make a difference?
 - Does it depend on the data / task?
- Symmetric Multiple-Negatives-Ranking-Loss
 - Used in CLIP Paper
 - Compute: $(\text{Loss}(A, P) + \text{Loss}(P, A)) / 2$
 - Swap anchor & positives (e.g. given answer, what is the question?)
 - Unclear impact

Multiple-Negatives-Ranking-Loss with Additive Margin

$$\blacksquare \text{sim}(a_i, p_j) = \begin{cases} \text{sim}(a_i, p_i) - m & \text{if } i = j \\ \text{sim}(a_i, p_j) & \end{cases}$$

- Subtract value m from positive pairs
 - Cosine-similarity with margin 0.3 used in LaBSE paper with translation pairs
- Unclear impact of margin for other tasks / datasets

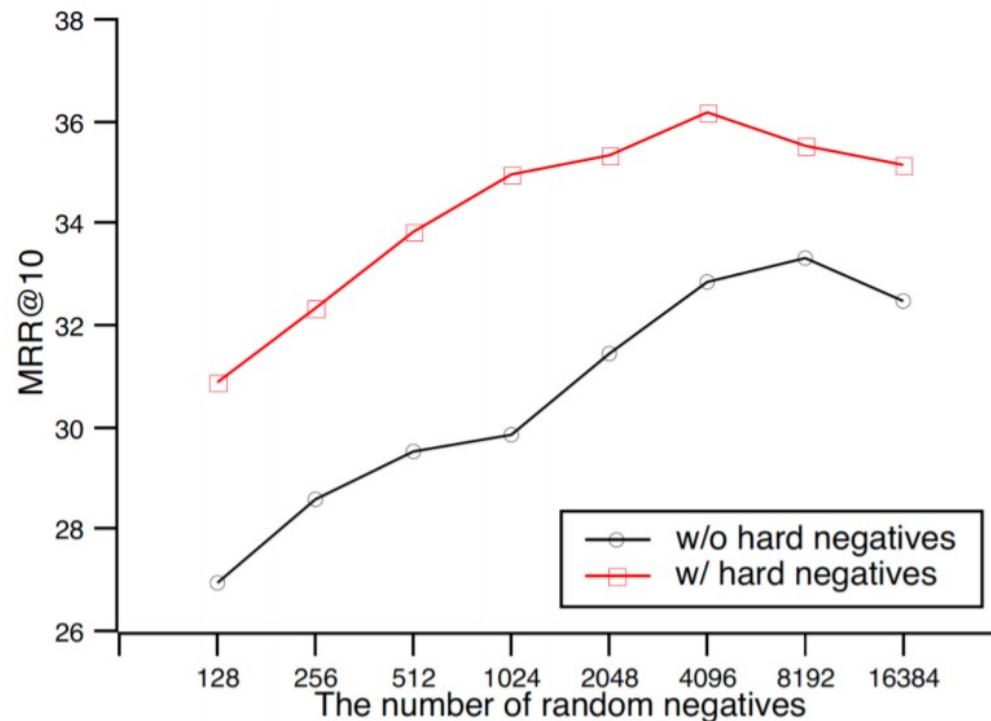
- Used in: LaBSE: <https://arxiv.org/abs/2007.01852> & <https://arxiv.org/abs/1902.08564>



Multiple Negative Ranking Loss

Hard Negatives

- Larger batch size => task more difficult => better results
 - Given query, which of the 10 passages provide the answer?
 - Given query, which of the 1k passages provide the answer?



Multiple Negative Ranking Loss

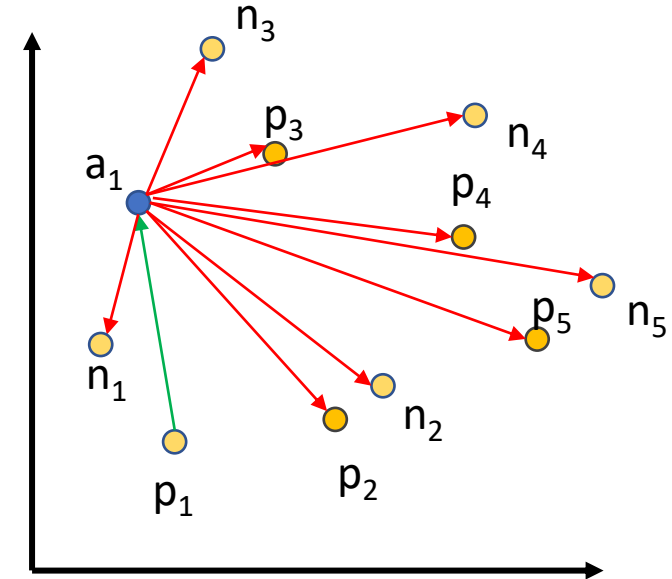
Hard Negatives

- Train with tuples:

(a_1, p_1, n_1)

(a_2, p_2, n_2)

- n_i should be similar to p_i but not match with a_i
- Bad example:
 - a: How many people live in London?
 - p: Around 9 million people live in London
 - n: London has a population of 9 million people.
- Good example:
 - a: How many people live in London?
 - p: Around 9 million people live in London
 - n: Around 1 million people live in Birmingham, second to London.



How to find hard-negatives?

- Quality of hard-negatives significantly improves the performance
- Finding good hard negatives not easy
- Strategy 1: Exploit structure in your data
 - Citation graph: (Title, Cited_Paper, Paper_Cited_by_Cited_Paper)
 - Q&A: (Question, Answer with many stars, Answer with few stars)
- Strategy 2: Mine hard negative:
 - Use BM25 to find top-100 most similar texts to anchor / positive
 - Select one of these randomly
 - Make sure that these are actually negatives!

Improving Quality with Better Batches

- Assume you have (question, answer) pairs from StackExchange
 - 140 different subforums: StackOverflow, Travel, Cooking, ...
- Naïve approach:
 - Randomly sample data from all pairs:
[(question_python, answer_python),
 (question_visa, answer_visa),
 (question_pasta, answer_pasta)]
 - Finding the right answer for a given question is easy
 - Question Python => Take that one programming answer in the batch...

Improving Quality with Better Batches

- Assume you have (question, answer) pairs from StackExchange
 - 140 different subforums: StackOverflow, Travel, Cooking, ...
- Better approach
 - Sample pairs from one subforum (e.g. StackOverflow)
 - [(question_python, answer_python),
 (question_java, answer_java),
 (question_c, answer_c)]

Improving Quality with Better Batches

- Assume you have (question, answer) pairs from StackExchange
 - 140 different subforums: StackOverflow, Travel, Cooking, ...
- Even better approach (?)
 - Sample pairs from same / similar tags (e.g. StackOverflow, Python tag)
 - [(question_python, answer_python),
 (question_numpy, answer_numpy),
 (question_pandas, answer_pandas)]
- Adding random batches might still be needed
 - Otherwise StackOverflow vector space could overlap with Travel vector space
 - 90% difficult batches, 10% easy random batches
 - Or: start with mainly random batches, then go to difficult batches

TPU Training Specialties

- TPU create a graph for your training steps
 - Graph is optimized & build.
 - Slow optimization & build step
 - Result is cached
- Tensors must all have the same length
 - Or use different padding length, e.g. 64, 128, 196, 256
 - Results in 4 graphs which are cached

Which datasets are suitable?

Name	Description	Size (#Pairs)	Performance
stackexchange_title_body_small	(Title, Body) pairs from different StackExchanges	364,001	59.83
gooaq_pairs	(Question, Answer)-Pairs from Google auto suggest	3,012,496	59.06
msmarco-query_passage_negative	(Query, Answer_Passage, hard_negative) from MS MARCO dataset	9,144,553	58.76
yahoo_answers_title_answer	(Title, Answer) pairs from Yahoo Answers	1,198,260	58.65
stackexchange_duplicate_questions	(Title, Title) pairs of duplicate questions from StackExchange	304,525	58.47
msmarco-query_passage	(Query, Answer_Passage) from MS MARCO dataset	532,751	58.28
eli5_question_answer	(Question, Answer)-Pairs from ELI5 dataset	325,475	58.24
yahoo_answers_title_question	(Title, Question_Body) pairs from Yahoo Answers	659,896	58.05
squad_pairs	(Question, Answer_Passage) Pairs from SQuAD dataset	87,599	58.02
yahoo_answers_question_answer	(Question_Body, Answer) pairs from Yahoo Answers	681,164	57.74
NQ-train_pairs	Training pairs (query, answer_passage) from the NQ dataset	100,231	57.48
quora_duplicates	Duplicate question pairs from Quora	103,663	57.36
WikiAnswers_pairs	Duplicate questions pairs from WikiAnswers	77,427,422	57.34
stackexchange_duplicate_questions	(Title+Body, Title+Body) pairs of duplicate questions from StackExchange	250,460	57.3
S2ORC_citation_pairs	Citation Pairs (Title, Title) of scientific publications from the S2OR corpus	52,603,982	57.28
stackexchange_duplicate_questions	(Body, Body) pairs of duplicate questions from StackExchange	250,519	57.26
quora_duplicates_triplets	Triples (question, duplicate_question, hard_negative) from Quora	103,663	56.97
AIINLI	Combination of SNLI + MultiNLI Triplets: (Anchor, Entailment_Text, Contradiction_Text)	277,230	56.57

Standard training setup of a MiniLM v6 model, 2k training steps, 256 batch size

Which models work well?

Model	Available in Flax / JAX?	Speed
MPNet	✗	2500
RoBERTa	✓	2500
BERT	✓	2500
DistilRoBERTa	✓ (?)	4000
TinyBERT-L6	✓	4500
DistilBERT	✗	4000
MiniLM-L12	✓	7500
MiniLM-L6	✓	14200
ALBERT	✓	2500
mBERT	✓	2500
XLNet	✗	2500

- Based on personal experience
- Ranked based on quality of derived embedding
- Better model != better model for embeddings (e.g. Electra performs rather bad)

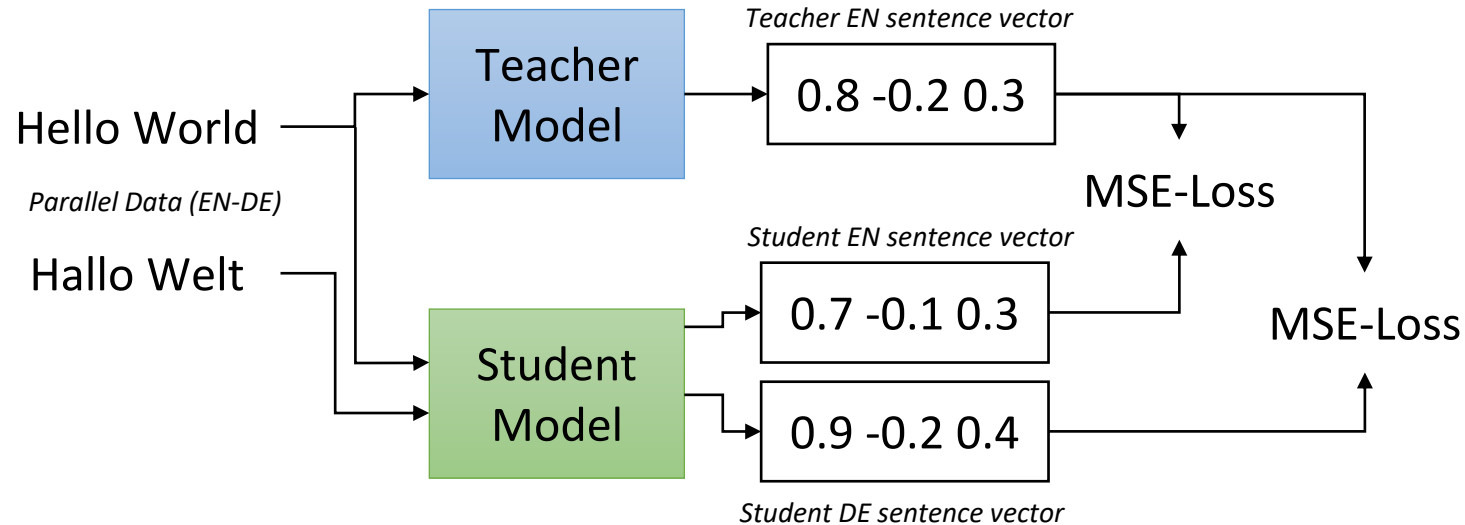
Multi-Dataset Training

- How to train with datasets of different sizes?
 - Natural Questions: 100k data pairs
 - Reddit: 600M+ data pairs
- Option 1: Equal sampling from all datasets
 - 50% NQ, 50% Reddit
- Option 2: Scaling with some temperature (see NMT literature)
- Option 3: Define a max data size, e.g. 1M
 - For Reddit, assume the size is just 1M
 - NQ: 100k
 - => 10% NQ, 90% Reddit
- Add cross dataset batches?

How to train Multilingual Models

- 1) Have suitable training data in your languages,
e.g. (question, answer)
- 2) Train with translation bridging task:
 - Have English data (e.g. (question, answer))
 - Have parallel data (e.g. (english_sentence, german_sentence))
 - Train alternating on English & parallel data
 - Was used for: mUSE

Multilingual Knowledge Distillation



- Given:
 - Teacher sentence embedding model T (e.g. SBERT trained on English STS)
 - Parallel sentence data $((s_1, t_1), \dots, (s_n, t_n))$
 - Student model S with multilingual vocabulary (e.g. XLM-R + Mean Pooling)
- Train student S such that:

$$S(s_i) \approx T(s_i)$$

$$S(t_i) \approx T(s_i)$$

Multilingual Knowledge Distillation

- Advantages:

- Easy to extend models later to more languages
- Training setup is easier: Focus on good English model, then make it multilingual
- Works on the general text domain as good / better than alternative approaches

- Disadvantages:

- Ignores country / language specific properties
- People from Japan ask different question than people from US
- Authentic training pairs from the language required to learn this

Timeline

Summary - Timeline 📅

23.06. - Official announcement of the community week. Make sure to sign-up in [this](#) 585 Google form.

23.06. - 30.06. Participants will be added to an internal Slack channel. Project ideas can be proposed here and groups of 2-10 are formed. Read [this](#) 207 document for more information.

30.06. - Release of all relevant training scripts in JAX/Flax as well as other documents on how to set up a TPU, how to use the training scripts, how to submit a demo, tips & tricks for JAX/Flax, tips & tricks for efficient use of the 🧠 hub.

30.06. - 2.07. Talks about JAX/Flax, TPU, Transformers, Computer Vision & NLP will be held.

7.07. Start of the community week! Access to TPUv3-8 will be given to each team.

7.07. - 14.07. - The Hugging Face & JAX/Flax & Cloud team will be available for any questions, problems the teams might run into.

15.07. Access to TPU is deactivated and community week officially ends.

16.07. Deadline for each team to submit a demo.

Open-sourcely yours,

The 🧠, 🤖, 🦋, Flax and Cloud team

Next Steps

- Join the discord server
- See the spreadsheet
- Add info to team
- Join a subteam
- Contribute:
 - Collect data
 - Work on the data loader: Weighting of datasets, lazy loading, sub-dataset batches vs. full-dataset batches (e.g. for StackOverflow, Reddit)