School of Data Science
香港城市大學
City University of Hong Kong

Evaluating Pre-trained Models for Unsupervised Contrastive Learning in Semantic Textual Similarity: A Comparative Study

Name: LU Guangrui
Group: 7
SID: 54019889
April 2024

School of Data Science
香港城市大學
City University of Hong Kong

Table of Content

Introduction to Semantic Textual Similarity (STS)

Introduction to Semantic textual similarity (STS)

In real world, we are likely to have different expression, conveying the same message or idea.

- " The quick brown fox jumps over the lazy dog."
- " A fast, brown fox leaps over a sleepy dog."

Such texts are only different in their expression, but fundamentally underlying the same scenario.

Introduction to Semantic textual similarity (STS)

STS technique analyzes sentences and computes their similarity score.

It’s a quantization of the similarity of different expression of natural languages in human society.

Introduction to Semantic textual similarity (STS)

In practice, STS is used in various applications, such as:

- machine translation

  (to compare the original text and the translated version)

- information retrieval

  (to find documents that are similar in content)

- question-answering systems

  (to match a user's question with similar questions in a database).

Introduction to Semantic textual similarity (STS)

The way to quantize textual similarity is to use sentence embeddings (cosine similarity).

With sentence embeddings extracted from pre-trained encoder models, can we use them to see a satisfactory result of semantic texture similarity now?

Not really!

Because these embeddings, as high dimensional vectors, often exhibit an ***anisotropic distribution***, where vectors are not evenly dispersed throughout the space.

Instead, they tend to cluster around a few significant directions. **This could lead models to be over-sensitive to few significant dimensions and ignore others, which may also contain meaningful information.**

Introduction to Semantic textual similarity (STS)

How to make the vector space more isotropic rather than anisotropic?

School of Data Science
香港城市大學
City University of Hong Kong

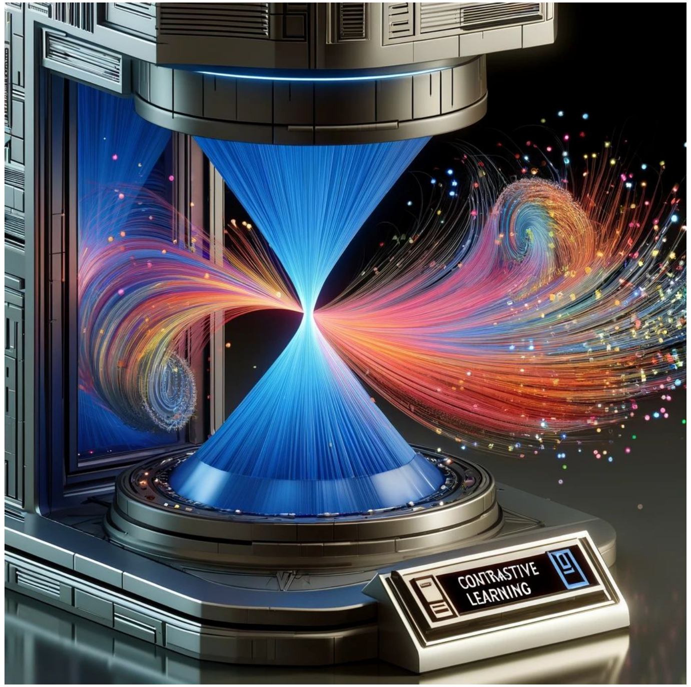
Introduction to Unsupervised SimCSE:

A simple contrastive learning of sentence embeddings

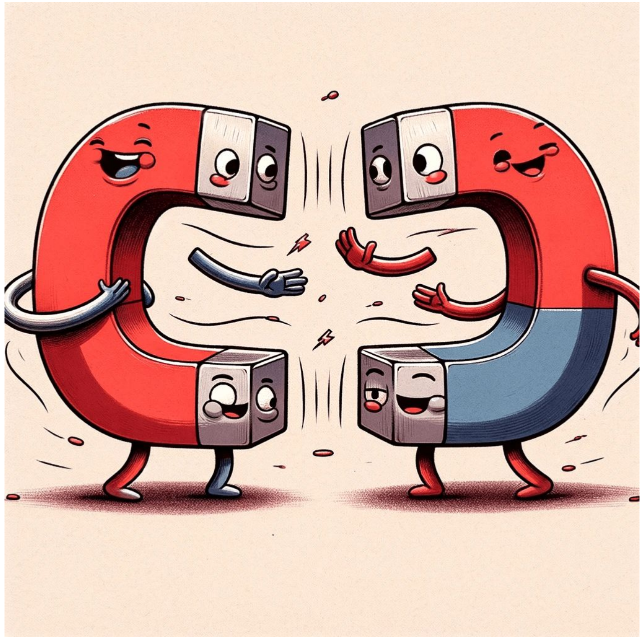
Introduction to unsupervised SimCSE

SimCSE introduces Contrastive Learning to make the vector space more isotropic rather than anisotropic.

It provides an extra regularization on the sentence embeddings, making them more uniformly aligned.



Introduction to unsupervised SimCSE: Contrastive Learning



In the field of unsupervised learning, **contrastive Learning** is a process where positive (similar) data pairs are pulled together and negative (dissimilar) pairs are pushed away.

Introduction to unsupervised SimCSE: Contrastive Learning

Contrastive Learning is optimized by the objective of contrastive loss, and a common example would be triplet loss (hinge loss).

$$L=\max(0,d(a,p)-d(a,n)+\alpha)$$

Where:

- $d(a,p)$ is the distance between the anchor and the positive sample.
- $d(a,n)$ is the distance between the anchor and the negative sample.
- α is the margin that is enforced between positive and negative pairs.

School of Data Science
香港城市大學
City University of Hong Kong

Introduction to unsupervised SimCSE: Loss

In unsupervised SimCSE (Gao.el, 2022), Contrastive learning is optimized by a different loss function, *softmax temperature-scaled cross-entropy loss*, which is a variant of InfoNCE (Oord.el, 2018).

$$L_i = -\log \frac{\exp(\mathrm{sim}(h_i, h_i^+)/\tau)}{\sum_{j=1}^{N} \exp(\mathrm{sim}(h_i, h_j)/\tau)}$$

- **hi** is the embedding of the anchor sentence input.
- ***hi+*** is the embedding of a "positive" sentence. Here for unsupervised case, it is from the same sentence with different dropouts.
- ***hj*** are embeddings of the negative sentences, i.e. other sentences in the mini-batch.
- **sim(,)** denotes cosine similarity.
- ***τ*** is a temperature parameter that scales the similarity scores.
- ***N*** is the number of sentences in the batch.

Introduction to unsupervised SimCSE: achievements

There are two major achievements in the unsupervised SimCSE:

1) Dropout outperforms other regularization methods, like Synonym replacement, word deletion, etc.

2) Contrastive Learning reduces the concentration of sentence embeddings, makes the vector space more uniformly aligned.

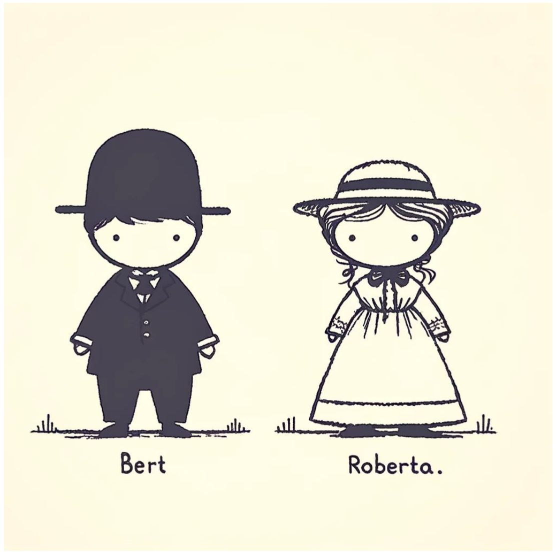
Experiments:

Replace pre-trained Models

Experiments: method

SimCSE implemented BERT and RoBERTa as its pre-trained LLM extractor for embeddings.

I added three more trending pre-trained LLMs, albert, Llama2 and Gemma.



Experiments: setup

Pre-trained Models	RoBERTa	BERT	ALBERT	Llama 2	Gemma
Training Set	1 million Wikipedia sentences				
Device	Google Compute Engine, Virtual Machine (single V100-16GB GPU)				
Batch size	512	64			
Learning rate	1e-5				
Temperature	0.05				

* Due to limited computational resource, all models get implemented to Contrastive Learning once, with one set of hyperparameters as shown above.

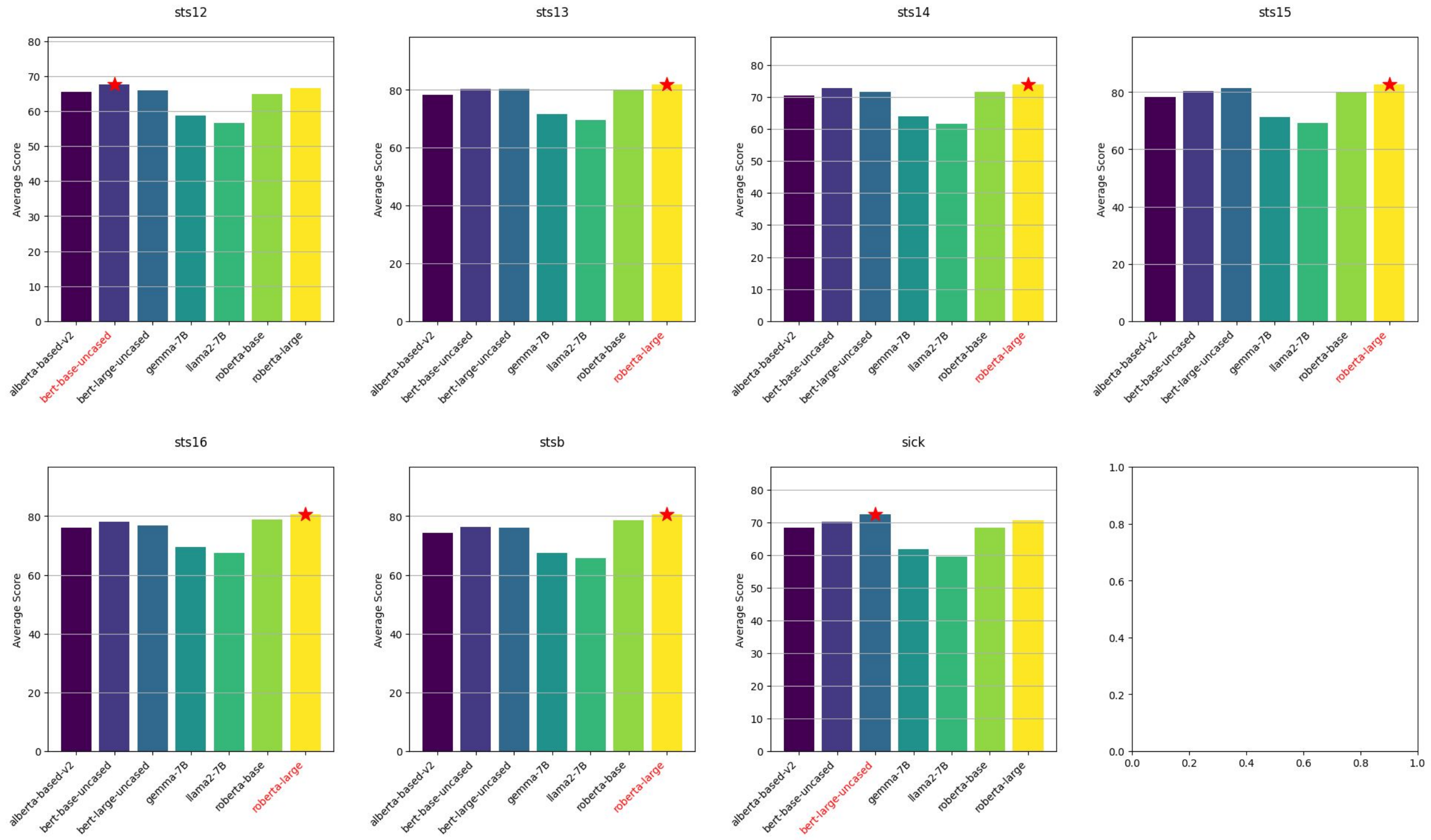
* The experiment is originally performed on colab pro, but colab does not give user full access to Rust setup, which is essential to build tokenizer with the tokenizer api from huggingface.

School of Data Science
香港城市大學
City University of Hong Kong

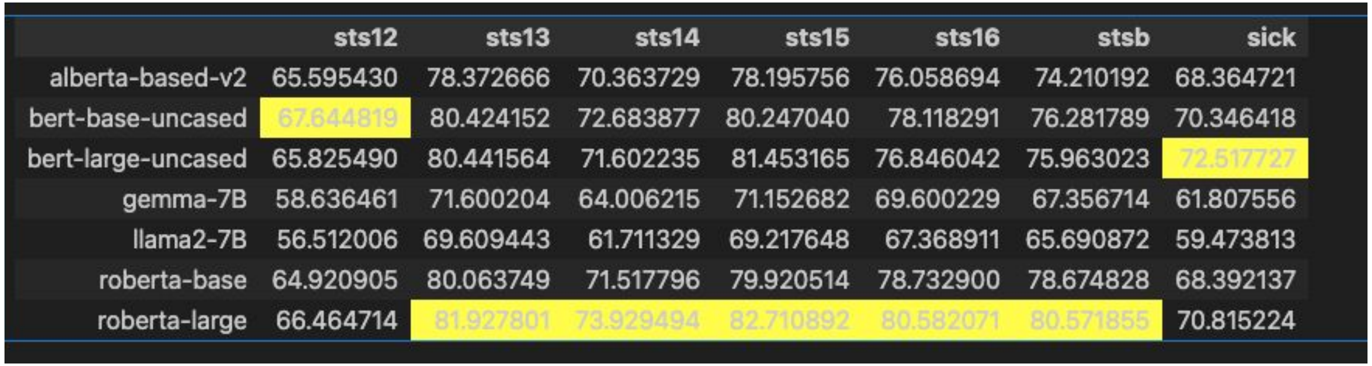
Results:

Performance && Robustness

Results: average performance



Results: average performance



	sts12	sts13	sts14	sts15	sts16	stsb	sick
alberta-based-v2	65.595430	78.372666	70.363729	78.195756	76.058694	74.210192	68.364721
bert-base-uncased	67.644819	80.424152	72.683877	80.247040	78.118291	76.281789	70.346418
bert-large-uncased	65.825490	80.441564	71.602235	81.453165	76.846042	75.963023	72.517727
gemma-7B	58.636461	71.600204	64.006215	71.152682	69.600229	67.356714	61.807556
llama2-7B	56.512006	69.609443	61.711329	69.217648	67.368911	65.690872	59.473813
roberta-base	64.920905	80.063749	71.517796	79.920514	78.732900	78.674828	68.392137
roberta-large	66.464714	81.927801	73.929494	82.710892	80.582071	80.571855	70.815224

School of Data Science
香港城市大學
City University of Hong Kong

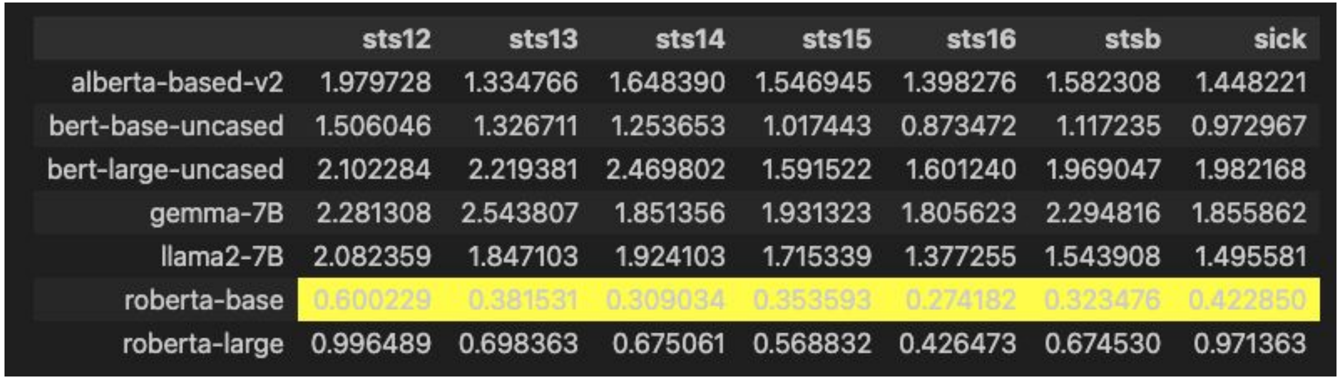
Results: average performance

1. RoBERTa-large unsupervised simCSE outperforms other models in most tasks in my experiments.

2. Masked Language Modeling (MLM) head API is used for BERT family, while CausalLM head API is used for non-BERT family. It might give BERT family some advantage regarding of their different optimization purpose.

3. Due to limited computational resources, only one set of hyperparameters is used for all models, this may prevent them to perform at their best.

Results: Standard Deviations



	sts12	sts13	sts14	sts15	sts16	stsb	sick
alberta-based-v2	1.979728	1.334766	1.648390	1.546945	1.398276	1.582308	1.448221
bert-base-uncased	1.506046	1.326711	1.253653	1.017443	0.873472	1.117235	0.972967
bert-large-uncased	2.102284	2.219381	2.469802	1.591522	1.601240	1.969047	1.982168
gemma-7B	2.281308	2.543807	1.851356	1.931323	1.805623	2.294816	1.855862
llama2-7B	2.082359	1.847103	1.924103	1.715339	1.377255	1.543908	1.495581
roberta-base	0.600229	0.381531	0.309034	0.353593	0.274182	0.323476	0.422850
roberta-large	0.996489	0.698363	0.675061	0.568832	0.426473	0.674530	0.971363

Results: Standard Deviations

RoBERTa-base unsupervised SimCSE has the least standard deviation in every single task, and thus is the most robust model.

The result aligns with its pre-trained models’ optimization goal.

Discussion:

Discussion: different models

Attribute	BERT	RoBERTa	ALBERT	LLAMA	GEMMA
Optimization/ Enhancements	Pre-training followed by fine-tuning for specific tasks	Dynamic masking	Parameter reduction techniques to increase training speed and reduce model size	Optimizations for faster computation	Integrated training across multiple tasks
Key Features	Uses masked language modeling (MLM) **Next sentence prediction (NSP)** Deeply bidirectional	**Removes NSP** Trains with much larger mini-batches and learning rates More data, longer training time	Factorized embedding parameterization Cross-layer parameter sharing **NSP replaced with sentence-order prediction (SOP)**	Designed for low-latency applications Focuses on efficient processing **Rotary Position** **Embeddings (RoPE)**	Multi-task learning from the ground up Simultaneously trains on various NLP tasks to generalize better **Rotary Position** **Embeddings (RoPE)**
Encoder Head	[CLS] token embedding from the last hidden state. Bidirectional layer		[CLS] and pooled output Bidirectional layer	**Unspecific** but generally used from Last hidden state Bidirectional layer most likely	

Discussion: Next Sentence Prediction(NSP)

Although NSP optimizes the contextual relationships between sentences, it seems beneficial only when long paragraphs are involved. **(EDM2021, J Feng, J Mostow)**

In our STS evaluation, without NSP, both Contrastive Learnings with albert and roberta achieves similar or better results compared to Contrastive Learning with BERT.

Discussion: Rotary Position Embeddings (RoPE)

- RoPE multiplies each token embedding by a rotation matrix that depends on its position.
- This integrates positional information directly with token embeddings in a dynamic manner.
- It should reduce the anisotropic drawback itself, by allowing similarity focused equally well on token embeddings as on relative position.

Discussion: Rotary Position Embeddings (RoPE)

However, the expected result is not shown on my experiment, with two possible reasons:

1. With one set of CL training and one set of hyperparameters, llama and Gemma does not reach its capability.
2. Contrastive Learning aligns the vectors in different dimensions too well and outcast the effect of RoPE.

Future Study:

Future Study

There are two possible ways to further improve the STS score with SimCSE.

1. We can try different set of hyperparameters for different pre-trained models, to reveal their true capability on extraction of embeddings.

2. We could try use larger pre-trained model to extract sentence embeddings.

School of Data Science
香港城市大學
City University of Hong Kong

References

Aaron van den Oord, Yazhe Li. Representation Learning with Contrastive Predictive Coding, Oriol Vinyals, 2018.

Jingrong Feng, Jack Mostow. Towards Difficulty Controllable Selection of Next-Sentence Prediction Questions, EDM 2021.

D. Cer, Y. Yang, S. yi Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, Y.-H. Sung, B. Strope, and R. Kurzweil. Universal sentence encoder, 2018.

A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes. Supervised learning of universal sentence representations from natural language inference data, 2018.

T. Gao, X. Yao, and D. Chen. Simcse: Simple contrastive learning of sentence embeddings, 2022.

R. Kiros, Y. Zhu, R. Salakhutdinov, R. S. Zemel, A. Torralba, R. Urtasun, and S. Fidler. Skip-thought vectors, 2015.

X. Li and J. Li. Angle-optimized text embeddings, 2023.

G. Majumder, P. Pakray, A. Gelbukh, and D. Pinto. Semantic textual similarity methods, tools, and applications: A survey. Computacion y Sistemas, 20(4):647–665, 2016.

Y. Yang, S. Yuan, D. Cer, S. yi Kong, N. Constant, P. Pilar, H. Ge, Y.-H. Sung, B. Strope, and R. Kurzweil. Learning semantic textual similarity from conversations, 2018.