

Report on Representation Contrastive Learning

Pablo Miralles^a (Researcher)

^aTechnical University of Madrid, Alan Turing st., Madrid, 28038, Spain

ARTICLE INFO

Keywords:

Contrastive Learning
Representation Learning
Self-supervised learning

ABSTRACT

This report delves into the emerging domain of contrastive learning, a powerful paradigm in deep learning that focuses on distinguishing between positive and negative examples to extract feature-rich representations of input data. An introduction to general representation learning is given, emphasizing its pivotal role in leveraging massive neural networks and overcoming data bottlenecks. We then introduce contrastive learning, and give an overview of several key areas of research in this field: the development of loss functions to effectively facilitate this learning approach; the strategies for generating data suitable for contrastive learning and the crucial role and challenges of utilizing negative examples, along with proposed solutions to these challenges. Finally, we study three applications of contrastive learning in creating state-of-the-art models. Through exploring these topics, this study aims to delineate the current advancements and applications of contrastive learning, providing a clear insight into its significance and potential for future exploration in the field of machine learning.

1. Introduction

In recent years, contrastive learning has emerged as a powerful paradigm in the field of deep learning. This approach, rooted in the fundamental idea of learning by contrasting positive and negative examples, has demonstrated remarkable success for learning feature-rich representations of input data across different modalities like image, audio, video or text. Further, it is used to build embeddings in metric spaces where a given notion of similarity has semantic meaning for humans.

The purpose of this text is to provide an introduction to the state of the art of the field of contrastive learning, as well as some of the most important topics within it.

In section 2 an introductory overview of representation learning is given, with a discussion on its advantages and disadvantages. Section 3 introduces contrastive learning, and establishes some notation and basic concepts for the rest of the text. Section 4 dives into the loss functions that are used in contrastive learning. Another important topic is how to generate data for contrastive learning, treated in section 5. In section 6, the topics of utilizing negative examples for learning is discussed in-depth. Finally, three state-of-the-art papers are explained in more detail in section 7.

2. Representation learning

One of the fundamental ideas of deep learning is the composition of transformations (layers). The output of each layer of the neural network can be seen as a different view or representation of the input data. Representation learning refers to the process of automatically discovering and extracting meaningful features or representations from the input data.

For classification or regression tasks, a final prediction head is added to the last layer of the neural network. After


training, one expects that this final layer of the network produces a representation of the data with meaningful features for the learned task.

The trend in recent years has been to train massive neural networks to learn very general representations of the input data, so that are useful for many different tasks. These models are trained using a complex *pretext task*. By changing the final head and maintaining the rest of the parameters, the model can later be *fine-tuned* to perform a different task, usually called *downstream task*. The process of *pre-training* with a different task is also called *transfer learning*. Contrastive learning has emerged as one of the most popular techniques for *pre-training* such models.

2.1. Why representation learning?

It is a natural question to wonder why bother with representation learning, instead of directly training for the downstream task. It has several advantages:

- For some downstream tasks there is not enough data, specially to train massive neural networks. Pretext tasks usually allow us to generate massive amounts of training data without effort. If the learned representation includes features that are useful for the downstream task, fine-tuning with just a few examples should be enough to achieve good results. Overcoming the training data bottleneck has allowed researchers to push model sizes and obtain state-of-the-art results for many tasks. This is the case of massive Transformer architectures like BERT Devlin, Chang, Lee and Toutanova (2019).
- Training massive networks requires incredible compute and energy resources, so much so that very few people can afford to train them. The appearance of open source models like BERT Devlin et al. (2019) and transfer learning has allowed many people to be

 pablo.miralles@upm.es (P. Miralles)
ORCID(S): 0000-0002-1230-9035 (P. Miralles)

able to access massive models. This is because fine-tuning while freezing the parameters of the pre-trained encoder requires much fewer resources.

- For downstream tasks where there is little or poor data, it is likely for models to learn spurious correlations to make good predictions. Due to the amount of diverse data and complexity of pretext tasks, we expect the model to have a general understanding of the underlying distribution of the input data, and not learn spurious correlations. Further, we also hope the fine-tuned model will generalize better than models just trained on the downstream task. It should be noted, however, that it is possible to worsen generalization during fine-tuning.
- They exhibit zero-shot capabilities. Large generative language models like GPT can perform a wide variety of zero-shot tasks, like classification. They obtain even better performance with few-shot learning, providing a few examples of the task in the input context (Brown, Mann, Ryder, Subbiah, Kaplan, Dhariwal, Neelakantan, Shyam, Sastry, Askell, Agarwal, Herbert-Voss, Krueger, Henighan, Child, Ramesh, Ziegler, Wu, Winter, Hesse, Chen, Sigler, Litwin, Gray, Chess, Clark, Berner, McCandlish, Radford, Sutskever and Amodei (2020)). CLIP (Radford, Kim, Hallacy, Ramesh, Goh, Agarwal, Sastry, Askell, Mishkin, Clark, Krueger and Sutskever (2021)), a model trained to learn which text caption best describes an image, can be used in arbitrary classification tasks by using as caption a description of the label.

Representation learning also has its disadvantages. First, the amounts of compute and energy required for training are enormous, and few people have access to such resources. Second, zero-shot performance is not that great of an advantage, as it often produces much worse performance than specialized models for a given task, at least without fine-tuning.

2.2. Evaluation of representation learning models

Since this work treats contrastive learning in the context of representation learning, the topic of evaluation should be addressed. Different techniques for representation learning use different pretext tasks, which means that the performance on these tasks is not a good way to evaluate them. We find two desirable conditions for representations. First, they should perform well for a particular downstream task of our interest. Second, we would like them be general and applicable to many problems without a great need of fine-tuning. Thus, the standard way to evaluate these methods will be to test the models on different downstream tasks.

3. Contrastive learning: basic concepts

Roughly speaking, in contrastive learning a vector representation of the input data is learned by comparing and

contrasting different examples. For instances that we deem similar in some sense, we want their vector representation to be close according to some metric. On the other hand, the vector representation of dissimilar examples should be further apart. In the case of images, we might treat pictures of the same concept (such as "dog") as similar.

Let's establish a framework and notation for contrastive learning for the rest of the text. It is a simplified version of that of Le-Khac, Healy and Smeaton (2020).

- We first consider encoders of the input data onto some vector representation space. The encoders are parameterized neural networks mapping examples to d -dimensional vectors $e_k(\cdot; \theta_k) : \mathcal{X}_k \rightarrow \mathbb{R}^d$.

It should be noted that we might have different encoders or not. For example, Radford et al. (2021) try to find common representations for images and text, where text that describes the image accurately should be close to the image itself. In this case, they use a different encoder for image and text. On the other hand, Reimers and Gurevych (2019) learn representations for text only, and when comparing two examples as similar or dissimilar, they are encoded through the same network, with tied weights. These are known as *siamese networks*.

- Additionally, we can assume without loss of generality that a final head maps these representations to a metric m -dimensional space $h_k(\cdot; \eta_k) : \mathbb{R}^d \rightarrow \mathbb{R}^m$, where the notion of distance or similarity is actually applied. The addition of a separate transformation to map the vector representation onto a different metric space separates the task of learning meaningful features and forcing similar instances to be close together. Prior to this addition, many methods found that intermediate layers of the trained encoder were superior for transfer learning than the final layer (Le-Khac et al. (2020)). Notice that if a method didn't need such a head, we might assume that these transformations are simply the identity function.
- The metric is given by a distance or similarity function $\mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}$, such as the Euclidean distance $\|z_1 - z_2\|$ or the cosine similarity $\frac{\langle z_1, z_2 \rangle}{\|z_1\| \|z_2\|}$. A similarity function takes larger values for similar examples, and will be denoted by s , whereas distance functions, denoted by d , are bounded by zero and take smaller values for similar instances.
- A loss function \mathcal{L} is built using this similarity notion for a set of encoded examples $\{h_{k_i}(e_{k_i}(x_i))\}_{i=1}^n$, penalizing large distances for similar instances, and small distances for dissimilar examples. Loss functions are further discussed in section 4.
- Finally, given an example $x \in \mathcal{X}$, we will denote by $v = e(x)$ its vector representation, and by $z = h(v)$ its embedding onto the metric space. Instances that we

consider similar to an anchor example x will be called positive examples and be denoted by a plus superscript x^+ . We define negative examples analogously and denote them by a minus superscript x^- .

4. Contrastive losses

There is a long history of loss functions used for contrastive learning. The aim of this section is to provide an overview and discussion of some of these functions, both the classical ones and the more modern ones used to train state-of-the-art models.

Let $x \in \mathcal{X}$ be an example and $z = h(e(x))$ its embedding in the metric space. The *pair loss* is defined for pairs of examples, differing for positive and negative examples (Chopra, Hadsell and LeCun, 2005):

$$\begin{cases} \mathcal{L}(x, x^+) = D(z, z^+)^2 \\ \mathcal{L}(x, x^-) = \max(0, \varepsilon - D(z, z^-))^2, \end{cases} \quad (1)$$

where D is some distance function such as the Euclidean distance. By minimizing this expression, the distance to positive samples should be close to zero, and dissimilar instances should be separated by a margin of at least ε .

The *triplet loss* requires instead that the distance with a positive example and the distance with a negative example differ at least by a margin ε (Schroff, Kalenichenko and Philbin, 2015):

$$\mathcal{L}(x, x^+, x^-) = \max(0, D(z, z^+) - D(z, z^-) + \varepsilon). \quad (2)$$

In this last loss function, a full training example requires now both a similar and a dissimilar instance. However, the interaction between different examples is still very limited. It has been shown theoretically and empirically that the use of negative examples, specially those that are hard for the model, are crucial for a good and efficient learning process (Le-Khac et al., 2020). Further discussion on the importance and problems of using negative samples is found in section 6. Here, it suffices to know that using several negative examples at once is more informative for the model and increases the chances of drawing examples that are hard.

Increasing the number of interactions between instances at once, Song, Xiang, Jegelka and Savarese (2016) proposed the *Lifted Embedding loss*. Let $\{x_i\}_{i=1}^n$ be a set of examples, and P and N the set of pairs of examples that are considered similar and dissimilar, respectively. Then the loss function is given as

$$\mathcal{L}(N, P) = \frac{1}{2|P|} \sum_{(i,j) \in P} L_{i,j}^2, \quad (3)$$

where

$$L_{i,j} = D_{i,j} + \log \left(\sum_{(i,k) \in N} e^{\varepsilon - D_{i,k}} + \sum_{(j,l) \in N} e^{\varepsilon - D_{j,l}} \right),$$

and $D_{i,j} = D(z_i, z_j)$. The expression for $L_{i,j}$ is actually used because it is a smooth upper bound for

$$\hat{L}_{i,j} = D_{i,j} + \max \left(\max_{(i,k) \in N} \varepsilon - D_{i,k}, \max_{(j,l) \in N} \varepsilon - D_{j,l} \right),$$

so this loss can be interpreted as an adaptation to the triplet loss, trying to mine the hardest negative example of the set for each positive pair, and squaring after the difference of distances. As usual in deep learning, instead of using the full set of examples, the loss is approximated with a batch of smaller size.

Let's now take a probabilistic perspective. If $P(\cdot|x_1, x_2)$ is the Bernoulli probability distribution function of x_1 and x_2 being similar, we might approximate this distribution as

$$P(1|x_1, x_2) = \sigma(s(z_1, z_2)), \quad (4)$$

where σ is the sigmoid function. If $p^+(\cdot, \cdot)$ and $p^-(\cdot, \cdot)$ are the probability distribution functions of similar and dissimilar instances, respectively, then the Binary Noise-Contrastive Estimation (NCE) loss (Gutmann and Hyvärinen, 2010) is given by minimizing the expected negative log-likelihood:

$$\begin{aligned} \mathcal{L}_{Bin-NCE} = & -\mathbb{E}_{p^+} \log P(1|x_1, x_2) \\ & - \mathbb{E}_{p^-} \log(1 - P(1|x_1, x_2)), \end{aligned} \quad (5)$$

where the expected value would be approximated by its population mean (e.g. with a single batch). Keeping the previous notation, this yields:

$$\begin{aligned} \mathcal{L}_{Bin-NCE} = & -\frac{1}{|P|} \sum_{(i,j) \in P} \log \sigma(s(z_i, z_j)) \\ & - \frac{1}{|N|} \sum_{(i,j) \in N} \log(1 - \sigma(s(z_i, z_j))). \end{aligned} \quad (6)$$

A more recent loss function is InfoNCE (van den Oord, Li and Vinyals, 2019). In their setting, instead of considering a binary classification problem they assume a ranking one. Having fixed an instance x , let $S = \{x_0^+, x_1^-, \dots, x_n^-\}$ be a set of possible similar instances, where only x_0^+ is a positive example. The problem becomes one of ranking which example is more likely to be positive. The probability of each of the samples can be approximated by a softmax operation on a similarity score with respect to x :

$$P(i|x, S) = \frac{\exp(s(x, x_i))}{\sum_{j=0}^n \exp(s(x, x_j))}. \quad (7)$$

Minimizing the negative log-likelihood of the true positive yields:

$$\mathcal{L}_{InfoNCE} = -\mathbb{E} \log \frac{\exp(s(x, x_0^+))}{\sum_{j=0}^n \exp(s(x, x_j))}. \quad (8)$$

As we will see, it is a common setting to have batches of pairs of similar instances $\{(x_1, x'_1), \dots, (x_n, x'_n)\}$, where

instances across pairs are considered to be dissimilar. We can compute a similarity matrix $\mathcal{S} = (s(z_i, z'_j))_{i,j}$, where the main diagonal values should be high and the rest should be low. In this case, one can calculate the InfoNCE across rows or columns. It is also possible to average both options, yielding a *symmetric InfoNCE* loss.

A temperature parameter τ can be included in the softmax operation (see e.g. (Chen, Kornblith, Norouzi and Hinton, 2020)), transforming Equation 7 into

$$P(i|x, \mathcal{S}) = \frac{\exp(s(x, x_i)/\tau)}{\sum_{j=0}^n \exp(s(x, x_j)/\tau)}. \quad (9)$$

A small value of τ makes the softmax sharper, and small differences between the similarity of positive and negative examples already produces a high likelihood. A large value of τ forces the difference in similarity to be large. This parameter can be viewed as the margin parameter in previous functions. This modified InfoNCE loss is termed *NT-Xent* (normalized temperature-scaled cross entropy loss) (Chen et al., 2020).

Some work has also been done with loss functions based on the concept of mutual information. A compilation of such work is left as a possible extension for the assignment for the ‘‘Seminars’’ class.

5. Data generation for contrastive learning

This section gives an overview of different ways to generate data for contrastive learning, based on (Le-Khac et al., 2020, Section III-B). While contrastive learning is generally agnostic to the supervision level paradigm, it is widely used for self-supervised learning. Thus, the techniques used to automatically generate data for a pretext task are of particular importance.

5.1. Human supervision

A possible approach to acquire data involves human annotation. For instance, Chi, Dong, Wei, Yang, Singhal, Wang, Song, Mao, Huang and Zhou (2021) leveraged sentence pairs in different languages to obtain cross-lingual language representations. In this context, sentences with similar meanings, representing translations of the same sentence, are deemed analogous, whereas sentences with distinct meanings are categorized as dissimilar. Radford et al. (2021) employed pairs comprising images and corresponding textual descriptions. Any remaining potential combinations of text and images are regarded as negative or non-analogous examples.

Securing extensive labeled datasets for this purpose is not always straightforward, particularly in the context of massive deep learning models. Under such circumstances, it is possible to resort to alternative forms of self-supervised pre-training for the encoders.

5.2. Data augmentation

One of the most common techniques in self-supervised learning is *data augmentation*. In the particular context of

contrastive learning, a transformation that does not change the instance semantically is applied to generate positive examples. To illustrate this, assume that we are contrasting images, and we want to map images based on the concepts inside them. Then if we crop, blur, or saturate the color of an image of a cat, it is still an image of a cat. This way, augmentations of similar or equal instances are also similar, whereas augmentations of dissimilar examples are also dissimilar.

To avoid any supervision, it is common to assume that each example in our dataset conforms its own class, and that it is dissimilar to all other instances. This particular task has been termed *instance-level discrimination* (Wu, Xiong, Yu and Lin, 2018). Of course, this might not always hold. We may have multiple pictures of similar cats in our dataset. We expect, however, that it is unlikely to draw many false negatives in a batch.

In the case of images, we find transformations such as rotations, translations, cutouts, cropping and resizing, blurring, applying noise... These techniques are applied in (Chen et al., 2020).

Textual data is a bit more complex. Fang, Wang, Zhou, Ding and Xie (2020) use automatic back-translation, that is, translation to a different language and then back to the source language, to generate pairs of different but semantically equivalent segments of text. Many other techniques can be used: changing characters, masking words, adding noise, changing words for synonyms... An extensive survey of method can be found in (Bayer, Kaufhold and Reuter, 2023).

Careful exploration of data augmentation techniques can be very important. For example, (Chen et al., 2020) found that the training of SimCLR was very sensitive to the choice of data augmentation techniques.

5.3. Other data generation techniques

There are several other approaches to generate data for contrastive learning. For example, one can capture different perspectives of the same information simultaneously. This is the case when using multiple camera angles to capture images at the same moment, or the combination of audio and images from the same video.

In the case of input data that can be decomposed as a series of local features, one can exploit the relationship between these features and a global representation of the instance. In this setting, the local and global representations of the same instance are deemed as similar, whereas those of different instances as dissimilar. For example, Deep InfoMax (Hjelm, Fedorov, Lavoie-Marchildon, Grewal, Bachman, Trischler and Bengio, 2019) was trained using this technique for images. A high level overview can be seen in subsection 5.3.

Finally, some data domains can be decomposed into a sequence of smaller units that have some consistency. In the case of videos, the images usually present some continuity throughout the time dimension. We can take images that are close in time as similar examples, and images that are far apart as dissimilar examples.

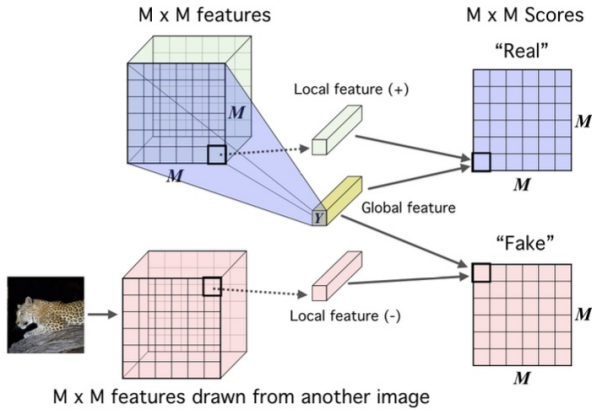


Figure 1: Production of positive and negative examples to train Deep InfoMax, figure taken from (Hjelm et al., 2019). The green box comes from an image of a dog, and the red box from an image of a leopard. The local features of the dog are aggregated into a global representation. The local and global features of the dog are similar, and the global features of the dog and the local features of the leopard are dissimilar.

6. Discussion on negative examples

We have seen in section 4 that most loss functions use both positive and negatives examples. There is a simple theoretical reason for this: if negative examples were not used, a trivial representation mapping everything to one vector would obtain perfect performance.

Further, there is empirical evidence that performance is increased from comparison to many negative examples (Le-Khac et al., 2020). For example, Chen et al. (2020) found that training with very large batch sizes greatly improved performance. In their setting, the batch size was tied to the number of negative examples for each positive example.

There are several topics discussed around the usage and generation of negative examples in contrastive learning, and this section provides an overview of some of them.

6.1. False negatives in self-supervised contrastive learning

In section 5, we already hinted at the possibility of false negatives when there is no supervision. This is because we are drawing from the full distribution of examples instead of the distribution of negative examples, creating a bias. Some work has been done on correcting this while not requiring manual labels for negatives. For example, Chuang, Robinson, Lin, Torralba and Jegelka (2020) proposed the *Debiased Contrastive loss*, sampling even more positive examples to correct this bias. The Debiased Contrastive loss improved performance in the presence of false negatives, but increased the computational cost of the loss function by also requiring several samples of similar instances.

6.2. Alleviating hardware bottlenecks for large amounts of negative samples

We have seen the use of examples in the same batch as negatives. This presents a major drawback in terms of computing resources. If we want to use many negative examples, we are forced to use a very large batch size. Some work has been done on decoupling batch size and the number of negatives by sampling negatives from an *offline memory bank*. In this setting, an encoded representation is kept on-disk for some or all examples, and the loss function is not back-propagated through them. The main difference between approaches lies in the method to keep these offline representations updated as the encoder is optimized.

Wu et al. (2018) sampled negative representations randomly from a memory bank with the full dataset. As the encoder is updated, the samples in the memory bank get outdated. At the end of each epoch, all the representations in the memory bank are updated with the new checkpoint of the model.

He, Fan, Wu, Xie and Girshick (2020) proposed keeping a queue with a fixed number of mini-batches. After processing a mini-batch, the new examples are added to the queue, and the oldest mini-batch is removed. The queue is used to sample negative examples for the current mini-batch. Since this alone resulted in poor empirical performance, they separated the online encoder that is being trained from an offline encoder that produces the representations for the queue. The parameters of the offline encoder are updated through a momentum update rule with the parameters of the online one:

$$\theta_{off} \leftarrow \alpha \theta_{off} + (1 - \alpha) \theta_{on}, \quad \alpha \in [0, 1). \quad (10)$$

This smoother update yielded better empirical performance.

6.3. Hard negative mining

While increasing the number of negative examples has been observed to improve performance, this might be due to the increased probability of finding meaningful negatives to learn from. These examples should be hard for the model, that is, their embedding should lie fairly close to the considered instance despite being negative.

Some work has been done on using a more careful selection of negative examples. For example, Kalantidis, Sariyildiz, Pion, Weinzaepfel and Larlus (2020) proposed an approach where the closest negative examples are drawn. They use feature space mixing techniques to generate even more negative examples.

However, hard negative mining has some disadvantages, such as the increased time complexity from sampling close neighbours and the increased probability of drawing false negatives in the self-supervised setting as the encoder gets better.

6.4. Are negative examples really necessary?

If the only reason to use negative examples is to prevent the representation from collapsing onto one single vector,

then other techniques to prevent it could allow us to remove negative examples.

Of particular importance seems to be the work by Grill, Strub, Alché, Tallec, Richemond, Buchatskaya, Doersch, Avila Pires, Guo, Gheshlaghi Azar, Piot, kavukcuoglu, Munos and Valko (2020). They used two neural networks, an online (predictive) network and a target network. They use only positive examples, and for those the online network tries to predict the metric representation from the target network. The parameters of the target network are updated after every iteration with an exponential moving average of the online parameters:

$$\theta_{target} \leftarrow \alpha \theta_{target} + (1 - \alpha) \theta_{online}.$$

It is not obvious to me that this approach would work. The authors argue that since the update to the target parameters is not exactly according to the gradient of the loss with respect to θ_{target} , there is no a priori reason to believe that the target network would converge to a collapsed representation. The authors did get good performance in downstream image classification tasks. There is some discussion on informal mediums (see blog post (Fetterman and Albrecht, 2020) and preprint (Richemond, Grill, Alché, Tallec, Strub, Brock, Smith, De, Pascanu, Piot and Valko, 2020)) on whether batch normalization is the cause of preventing representational collapse, but I haven't found peer-reviewed work on the topic.

7. Applications

In this section, we explore how contrastive learning is used in producing state-of-the-art models through a more detailed explanation of three seminal papers.

7.1. Sentence-Bert

Reimers and Gurevych (2019) introduced Sentence-BERT. BERT (Devlin et al., 2019) is a language encoder-only transformer-based model. It learned a representation of the input text not by contrastive learning, but by two pretext tasks known as Masked Language Modeling and Next Sentence Prediction. BERT admits pairs of texts as input, which allows it to perform tasks such as Semantic Textual Similarity or Natural Language Inference on the two input texts. However, when it is necessary to perform a task for each pair in a large set of texts, the time complexity of executing the model becomes prohibitive.

In contrast, Sentence-BERT overcomes this by using a BERT encoder to obtain a vector representation out of each of the input texts. The same network (with tied weights) is applied to each text, and the individual representations are combined in a simple head to obtain the output. They used the following heads and objective functions, for pairs of representations u and v :

- **Classification objective.** A linear transformation of the concatenated $(u, v, |u - v|)$, combined with a softmax layer for classification. A classification loss can then be applied.

- **Regression objective.** Cosine similarity of u and v .
- **Triplet objective.** As seen in section 4.

They fine-tuned the BERT encoder with these objectives on downstream tasks with supervised data. The obtained representations had great performance on Semantic Textual Similarity. With this approach, the task of finding the most similar sentence pair in a set of 10000 sentences from 65 hours to approximately 5 seconds. Instead of computing pair-wise embeddings, they compute a single embedding for each text and compare them using cosine-similarity. Further, the obtained embeddings for each text can take advantage of advanced index structures for fast k-NN approximations. However, for more complex tasks such as Natural Language Inference, the performance is worse than the original BERT approach.

7.2. SimCLR

Chen et al. (2020) showed a state-of-the-art contrastive learning approach for image representation learning.

They used a ResNet-50 deep learning architecture to obtain the representation of the image, followed by a small MLP network of one hidden non-linear layer to produce the embedding in a metric space. In the metric space, the NT-Xent loss from section 4 is used. By applying data augmentation, they obtain two similar images from the original one. They considered every other image in the batch to be dissimilar. A diagram of this approach is shown in subsection 7.2.

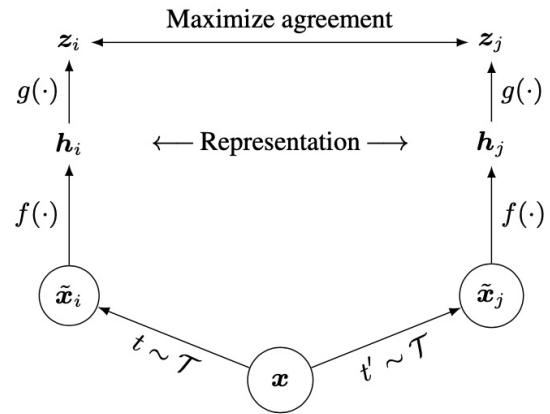


Figure 2: Diagram of the SimCLR training approach, taken from (Chen et al., 2020). \mathcal{T} is the data augmentation function. Each example then goes through a ResNet-50 encoder f , and then a final projection MLP head g . In the final metric space, the agreement is measured.

The key ingredients were:

- **Strong data augmentation.** They found that results were very sensitive to the data augmentation strategy. Particularly effective was the composition of random crop and resize, color distortions and Gaussian blur.

- **Use of a projection head.** The obtained representations from the main architecture (ResNet) were much better than without the projection head, that is, using the representation space as a metric space as well.
- **Large batch size.** Large batch sizes were very important for performance, specially with a small number of training epochs. The best results seem to be obtained from batches of 4096, that is, 8192 negative examples per batch.

With this approach, they achieved state-of-the-art performance in self-supervised, semi-supervised (using 1% and 10% of labeled data) and transfer learning. Further, a linear classifier trained on the representations of SimCLR obtained competitive performance with fully-supervised models on ImageNet, albeit with four times the model size.

7.3. CLIP

Radford et al. (2021) proposed CLIP. They jointly trained an image encoder (ResNet or a Vision Transformer) and a text transformer-based encoder. They created a dataset of 400 million pairs of images and captions describing the images. For a batch size of N pairs, they trained using the symmetric InfoNCE loss. Each batch contained N positive pairs of corresponding images and captions, while the $N^2 - N$ remaining possible pairs are considered negative.

The text interface due to training with natural language is the best feature, since it allows zero-shot application to many classification datasets. By comparing with textual labels we can perform multi-class classification on arbitrary classes.

On image classification, CLIP achieves competitive performance with a fully supervised, regularized, logistic regression classifier on the representation generated by a ResNet50 architecture. This is well below state-of-the-art performance, but it is nonetheless impressive given that no fine-tuning was involved. The worst performance is obtained, as expected, on very niche categories such as lymph node tumor detection. CLIP appears to be able to perform tasks like geo-localization, Optical Character Recognition, facial emotion recognition and action recognition.

8. Conclusions

Contrastive learning has solidified its position as a potent paradigm in the realm of deep learning, especially in the extraction of feature-rich representations across various modalities such as images, audio, video, and text. We have seen many advantages of representation learning in general: the ability to overcome scarcity of data for downstream tasks, the ability to train massive deep learning models that generalize well and do not learn spurious correlations, the zero-shot capabilities... For contrastive learning in particular, we have seen with Sentence-BERT the performance advantage for certain tasks of learning semantic embeddings in metric spaces. However, we still see many challenges and limitations of these techniques:

- **Resource Intensity:** One of the most significant challenges is the enormous amounts of data, compute, and energy required for training. Such resource intensity makes it inaccessible for many. More particularly, contrastive learning benefits from large batch sizes, posing even bigger hardware challenges than other techniques for representation learning.
- **Zero-shot Performance:** While models like CLIP exhibit zero-shot capabilities, their performance is often subpar compared to specialized models, especially without fine-tuning.
- **Design Sensitivity:** The training of models can be very sensitive to design parameters. For example, SimCLR is highly sensitive to the choice of data augmentation techniques.
- **False Negatives:** In self-supervised contrastive learning, the assumption that each example belongs to its unique class can lead to the challenge of false negatives, as we have seen. Existing solutions have not been widely adopted to the best of my knowledge. This is probably due to the increased complexity and reduced performance.

Contrastive learning has reshaped representation learning, but it's not without challenges. Its benefits and limitations highlight the need for ongoing research. As the deep learning field evolves, it's crucial for the community to address these challenges, ensuring that contrastive learning remains impactful and relevant in the broader AI landscape.

References

- Bayer, M., Kaufhold, M.A., Reuter, C., 2023. A Survey on Data Augmentation for Text Classification. *ACM Computing Surveys* 55, 1–39. doi:10.1145/3544558.
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D., 2020. Language Models are Few-Shot Learners, in: *Advances in Neural Information Processing Systems*, Curran Associates, Inc.. pp. 1877–1901.
- Chen, T., Kornblith, S., Norouzi, M., Hinton, G., 2020. A Simple Framework for Contrastive Learning of Visual Representations, in: *Proceedings of the 37th International Conference on Machine Learning*, PMLR. pp. 1597–1607.
- Chi, Z., Dong, L., Wei, F., Yang, N., Singhal, S., Wang, W., Song, X., Mao, X.L., Huang, H., Zhou, M., 2021. InfoXLM: An Information-Theoretic Framework for Cross-Lingual Language Model Pre-Training, in: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, Online. pp. 3576–3588. doi:10.18653/v1/2021.naacl-main.280.
- Chopra, S., Hadsell, R., LeCun, Y., 2005. Learning a Similarity Metric Discriminatively, with Application to Face Verification, in: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, IEEE, San Diego, CA, USA. pp. 539–546. doi:10.1109/CVPR.2005.202.

- Chuang, C.Y., Robinson, J., Lin, Y.C., Torralba, A., Jegelka, S., 2020. Debaised Contrastive Learning, in: *Advances in Neural Information Processing Systems*, Curran Associates, Inc.. pp. 8765–8775.
- Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, in: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Association for Computational Linguistics, Minneapolis, Minnesota. pp. 4171–4186. doi:10.18653/v1/N19-1423.
- Fang, H., Wang, S., Zhou, M., Ding, J., Xie, P., 2020. CERT: Contrastive Self-supervised Learning for Language Understanding. doi:10.48550/arXiv.2005.12766, arXiv:2005.12766.
- Fetterman, A., Albrecht, J., 2020. Understanding self-supervised and contrastive learning with "Bootstrap Your Own Latent" (BYOL). <https://imbue.com/research/2020-08-24-understanding-self-supervised-contrastive-learning/#our-surprising-results>.
- Grill, J.B., Strub, F., Alché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., Piot, B., kavukcuoglu, k., Munos, R., Valko, M., 2020. Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning, in: *Advances in Neural Information Processing Systems*, Curran Associates, Inc.. pp. 21271–21284.
- Gutmann, M., Hyvärinen, A., 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models, in: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, JMLR Workshop and Conference Proceedings*. pp. 297–304.
- He, K., Fan, H., Wu, Y., Xie, S., Girshick, R., 2020. Momentum Contrast for Unsupervised Visual Representation Learning, in: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Seattle, WA, USA. pp. 9726–9735. doi:10.1109/CVPR42600.2020.00975.
- Hjelm, R.D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Bachman, P., Trischler, A., Bengio, Y., 2019. Learning deep representations by mutual information estimation and maximization, in: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, OpenReview.net.
- Kalantidis, Y., Sariyildiz, M.B., Pion, N., Weinzaepfel, P., Larlus, D., 2020. Hard Negative Mixing for Contrastive Learning, in: *Advances in Neural Information Processing Systems*, Curran Associates, Inc.. pp. 21798–21809.
- Kumar, P., Rawat, P., Chauhan, S., 2022. Contrastive self-supervised learning: Review, progress, challenges and future research directions. *International Journal of Multimedia Information Retrieval* 11, 461–488. doi:10.1007/s13735-022-00245-6.
- Le-Khac, P.H., Healy, G., Smeaton, A.F., 2020. Contrastive Representation Learning: A Framework and Review. *IEEE Access* 8, 193907–193934. doi:10.1109/ACCESS.2020.3031549.
- van den Oord, A., Li, Y., Vinyals, O., 2019. Representation Learning with Contrastive Predictive Coding. arXiv:1807.03748.
- Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I., 2021. Learning Transferable Visual Models From Natural Language Supervision, in: *Proceedings of the 38th International Conference on Machine Learning*, PMLR. pp. 8748–8763.
- Reimers, N., Gurevych, I., 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, in: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Association for Computational Linguistics, Hong Kong, China. pp. 3982–3992. doi:10.18653/v1/D19-1410.
- Richemond, P.H., Grill, J.B., Alché, F., Tallec, C., Strub, F., Brock, A., Smith, S., De, S., Pascanu, R., Piot, B., Valko, M., 2020. BYOL works even without batch statistics. arXiv:2010.10241.
- Schroff, F., Kalenichenko, D., Philbin, J., 2015. FaceNet: A unified embedding for face recognition and clustering, in: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Boston, MA, USA. pp. 815–823. doi:10.1109/CVPR.2015.7298682.
- Song, H.O., Xiang, Y., Jegelka, S., Savarese, S., 2016. Deep Metric Learning via Lifted Structured Feature Embedding, in: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Las Vegas, NV, USA. pp. 4004–4012. doi:10.1109/CVPR.2016.434.
- Weng, L., 2021. Contrastive representation learning. lilianweng.github.io.
- Wu, Z., Xiong, Y., Yu, S.X., Lin, D., 2018. Unsupervised Feature Learning via Non-parametric Instance Discrimination, in: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Salt Lake City, UT. pp. 3733–3742. doi:10.1109/CVPR.2018.00393.

A. Bibliography analysis

A.1. Methodology

To find relevant work about the topic I used primarily three secondary sources. The main one is the survey (Le-Khac et al., 2020). I also included some articles from the seminar by Javier Huertas-Tato and from the recommended reading from the seminar (Weng, 2021). After compiling a good amount of papers, I also used the ResearchRabbit and Inciteful applications to find relevant and related papers to the ones I already had, without much success.

A.2. Analysis

In tables 1 to 3 we find the different sources cited in the report as credible, together with some impact metrics. For journal articles I have included the JCR quartile of the journal (at the time of publishing) and the number of citations. For conferences, the core rank (at the earliest time after publishing) and number of citations are shown. Finally, for preprints without peer review I only included the number of citations. We observe the following.

- The secondary sources are the worst ones. The seminar and recommended reading are not peer-reviewed, and the survey was published in a Q2 journal and has moderate citations. However, I think that the followed methodology should be sufficient to discover most of the relevant literature, and the credibility of primary sources has been checked as well.
- Further, the used survey is from 2020, fairly old (considering the pace of progress in deep learning in the last few years). I have also checked a more recent survey (Kumar, Rawat and Chauhan, 2022), and used applications for literature discovery, without finding very relevant after 2021. This might indicate that the progress in this field has staled a bit.
- Of the used pre-prints without peer-review, one of them is highly cited (more than 6000 citations), and the other has moderate citations (256). In any case, the last one just served as an example for the use of back-translation, and was cited in a peer-reviewed survey.
- The remaining journal article has moderate citations but was published in a very relevant journal (Q1, 3 out of 111 in Computer Science, Theory & Methods).
- Of the conference papers, all of them are either published in A* conferences or have more than 1500

Table 1
Cited conference papers

Entry	Year	Conference	# citations	Core rank
Chi et al. (2021)	2021	Annual Conference of the North American Chapter of the Association for Computational Linguistics	245	A
Radford et al. (2021)	2021	International Conference on Machine Learning	8651	A*
Chen et al. (2020)	2020	International Conference on Machine Learning	12601	A*
Grill et al. (2020)	2020	Advances in Neural Information Processing Systems	4559	A*
Chuang et al. (2020)	2020	Advances in Neural Information Processing Systems	412	A*
Kalantidis et al. (2020)	2020	Advances in Neural Information Processing Systems	440	A*
Brown et al. (2020)	2020	Advances in Neural Information Processing Systems	15354	A*
He et al. (2020)	2020	IEEE Conference on Computer Vision and Pattern Recognition (CVPR)	8835	A
Devlin et al. (2019)	2019	Annual Conference of the North American Chapter of the Association for Computational Linguistics	80335	A
Hjelm et al. (2019)	2019	International Conference on Learning Representations (ICLR)	2414	A*
Reimers and Gurevych (2019)	2019	Conference on Empirical Methods in Natural Language Processing & International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)	7076	A, B
Wu et al. (2018)	2018	IEEE Conference on Computer Vision and Pattern Recognition (CVPR)	3097	A
Song et al. (2016)	2016	IEEE Conference on Computer Vision and Pattern Recognition (CVPR)	1735	A
Schroff et al. (2015)	2015	IEEE Conference on Computer Vision and Pattern Recognition (CVPR)	14997	A
Gutmann and Hyvärinen (2010)	2010	International Conference on Artificial Intelligence and Statistics	2110	B
Chopra et al. (2005)	2005	IEEE Conference on Computer Vision and Pattern Recognition (CVPR)	4780	A

citations, except for (Chi et al., 2021). This last paper has moderate citations (245) and was published in a Core rank A conference.

- Although I haven't performed an exhaustive analysis of authors, we see some big names such as Yoshua Bengio, Yann LeCun, Geoffrey Hinton, Alec Radford, Oriol Vinyals,...

In general, I think the selected primary sources are very reputable. We see many papers with an exceptionally high number of citations, and many published in top conferences such as NeurIPS.

Table 2

Cited journal articles

Entry	Journal	year	# citations	Quartile
Bayer et al. (2023)	ACM Computing Surveys	2023	171	Q1 (3/111 Computer Science, Theory & Methods)
Le-Khac et al. (2020)	IEEE Access	2020	381	Q2 (65/161 Computer Science, Information Systems)

Table 3

Cited ArXiv preprints

Entry	Year	# citations
Fang et al. (2020)	2020	256
van den Oord et al. (2019)	2018	6256