Representation Learning with Transformer-based Contrastive Predictive Coding

Murtadha Bahrani
Boston University
murtadha@bu.edu

Abstract

Learning a general representation for multimodal data can be useful in enhancing performance for downstream cross-modal tasks. Several methods have been proposed to tackle this issue such as the Contrastive Predictive Coding CPC [16]. The autoregressive module of CPC uses a GRU RNN model to encode latent representations into context representation vectors. The success of Transformer networks for NLP tasks motivates exploring its usage for unsupervised representation learning. In this project we introduce a Transformer-based CPC architecture. Furthermore, we apply our model on unlabeled audio data and test the learned representations on downstream tasks.

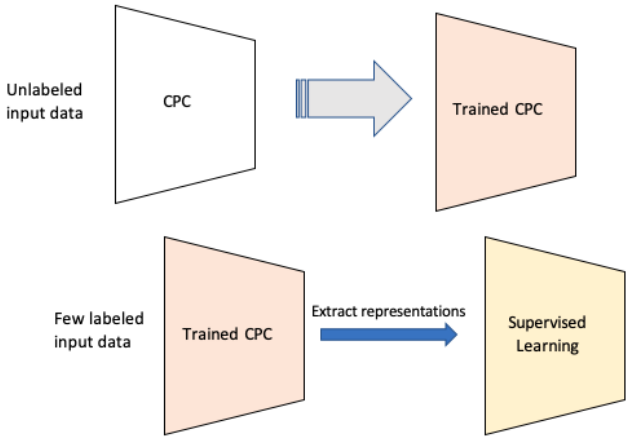

Figure 1. Example of using CPC.

1. Introduction

Supervised learning has proven successful for many tasks in different modalities. Its applications span from visual, language, and audio simple tasks, to more complex ones that include two or more modalities. However, its learned representations are specialized for each task and often transfer learning methods are used to generalize the model for other realted tasks. One of the proposed methods to tackle this issue is to use unsupervised learning to learn general representations and use the learned representations for downstream tasks. This approach is more data efficient since its learned weights for extracting general representations can be used for other tasks.

One recent method that showed promising results is Contrastive Predictive Coding (CPC) [16]. The CPC method comprises two main components, one to encode the input data into latent representations, and another to encode those representations into a context vector representation. It also leverages several ideas such as predictive coding, importance sampling, and contrastive loss to learn general representations. Since its appearance, several works built on some parts of it have achieved promising results [14, 9].

The main goal of CPC is to train the model to learn latent representations or context representations that are useful

downstream tasks. In other words, CPC is trained mainly to learn general representations rather than special representations for the given task. For the context vector representations, the GRU model [5] is used in CPC to encode the latent representations of the input data into context vector representations. So the latent representations here are temporal and hence other sequence-to-sequence models can be used. Since Transformers [17] have shown great success in language related tasks, due to its ability to encode informative context of long sequences, we are interested in investigating its use in learning general representations. We hypothesize that Transformer-based CPC architecture would produce representations that might be more informative for downstream tasks than GRU-based CPC. Our proposed modification to CPC is mainly concerned with the autoregressive model module. Instead of using a GRU with 256 hidden dimension, as in the original implementation of CPC, we use a Transformer architecture with 512 hidden dimension. Through our experiments we focus on one modality, audio, but base our architecture choices to be applicable for different modalities. While there are many implementations of Transformers for audio data, most of them depend on extracting the vocabulary from the training data. We avoided this approach because it's not aligned with the goal of this project, which is a universal method for different

modalities. For the Transformer block, we used similar architecture proposed in [11], in which a convolutional layer placed below the transformer block and served to extract relative positional information to discover local features at each time step. However, we hypothesis that the convolutional layer would not be necessary in our case since our convolutional encoder is already deep enough to learn local features. In summary, the proposed architecture uses Transformers with muli-head attention in the autoregressive module of CPC. Moreover, the model is trained end-to-end using contrastive loss and mini-batch negative sampling. Finally, through experiments we show that Transformer-based CPC is capable of learning general representations that perform well for downstream tasks compared to GRU-based CPC.

2. Related Work

Unsupervised Learning. This project builds on the idea that unsupervised learning can be used to learn representations that are efficient and robust for various modalities [6]. We rely on the work proposed in [16] that uses unsupervised learning to construct such representations. [16] show competitive performance in downstream supervision tasks for various modalities.

Predictive Coding. The main idea behind contrastive predictive coding is to learn high-level representation from high dimensional noisy data by predicting encoded latent representations [16]. Thus, both the latent representations and high-level representations are induced, though optimizing for contrastive loss, to capture both local and global features. This is also known as learning slow features, or global structure of the data. The CPC architecture has two main modules, an encoder to encode the input sequences into latent representations and an autoregressive model to encode context representations. Both modules are jointly trained, with negative sampling, to optimize the noise contrastive loss. The encoder part contains strided convolutional layers with ReLU activations. The autoregressive part uses a GRU RNN model [5].

Contrastive Loss. Instead of optimizing the model to minimize a loss function defined as the prediction error, contrastive loss function is defined as a distance function between data points or augmented views of the same data point [4]. Unlike prediction error based loss, contrastive loss does not require supervision given as ground truth labels. Several applications employing unsupervised contrastive learning have shown success comparable to or outperforming its supervision counterparts [3, 15].

Noise Contrastive Estimation. Noise contrastive estimation (NCE) is a technique that is used to estimate a density ratio function from generated noise [8]. Combining NCE with sampling techniques, such as mini-batch negative sampling, provides a tractable way to estimate Mutual Information (MI) between two representation vectors. [16] define the MI between context vector and input data as a density ratio function that is a simple log-bilinear model of a linear transformation. Subsequently, it uses NCE to estimate the weights of these linear transformations.

Transformers. Transformers are encoder-decoder network architecture that is used for sequence-to-sequence modelling. Since it was first introduced in [17], it had several successful applications and gained a towering reputation especially in language related tasks []. Transformers do not use recurrent or convolutional layers and rely only on the attention mechanism [2]. The encoder components consist of a multi-head self-attention layer and a feed forward NNs. The decoder consists of the same components with an extra attention layer. The purpose of the multi-headed attention layers is to capture different dependencies of each index of the sequence. This allows for parallelization and more content aware output sequences. Recent adaptations of Transformers to multi modal learning have been introduced. [7] apply CNN-free Transformers on sequences of image patches, called ViT, showing relatively excellent results compared to state-of-art CNN models. [11, 1] apply Transformers to speech recognition tasks and use a convolutional layer instead of a fixed positional embedding. For reinforcement learning, [13] deploy Transformers architecture variant, called GTrXL, on experience stored in memory to compute representations of the observations used to predict the value function.

2.1. Our Approach

Our proposed model is an adaptation of [16] consisting of two modules. We modify the second module to use Transformers instead of a GRU RNN model. In the following subsections we provide some details of each module. Let $X = x_t$ be the input data, e.g. x is a 1-second segment of an audio recording of someone telling a story. At each x there is something said about some events in the story. Moreover, at each x, there might be also some background noise like a kid talking about something else. A good representation of these recordings up until time t would be a context vector c_t that has information of $x_{\leq t}$, e.g. the storyline but not the noise. The goal is to maximize the mutual information between c_t and x_{t+k} for k time-step towards the future. In other words, we want to increase the predictive power of the context vectors c. Recall that CPC uses an encoder for the input data, that is a stack of convolutional layers, to produce latent representation $z_t = encoder(x_t)$. So rephrasing our objective, the goal turns to be maximizing the mutual information between c_t and $z_{\leq t}$, which is bounded by the MI of c_t and $x_{\leq t}$. Simply,

$$MI(c_t, z_{\leq t}) \leq MI(c_t, x_{\leq t}).$$

The Transformer module is used to encode sequences of latent representations into context vector representations

$c_t = Transformer(z_{\leq t})$.

The model is trained to maximize the mutual information between the context vector and input data x, by minimizing the difference between c_t and x_{t+k} for some k. However, as stated previously, the future observations x_{t+k} is not directly predicted, therefore a density ratio function is defined

$$f(c_t, x_{t+k}) = exp(z_{t+k}^T W_k c_t).$$

Simply, for each timestep k in the future, the weights W_k of a fully connected layer of context c_t are learned, and the similarity score to z_t is computed. Finally, the model is trained end-to-end to optimize a loss based on NCE

$$L_B = -\frac{1}{|B|} \sum_B \log(\frac{f_k(c_t, x_t + k)}{\sum_{x_i \in B} f_k(c_t, x_i)}),$$

where B is a minibatch containing one positive example and the rest are negatives.

2.2. The Encoder Module

The feature encoder consists of four blocks, each has 1D convolutional layer followed by ReLU activations and batch normalization.

2.3. The Transformer Module

We use the same Transformer variant proposed in [11] but without the convolutional layer placed below the block. It consists of two blocks, first, a multi-head layer and a dropout layer wrapped with a residual connection, then a layer normalization. Second, two fully connected layers with ReLU activation in between, wrapped with a residual connection, then finally a layer normalization.

3. Experiments

3.1. Dataset

For our experiments we used the LibriSpeech Corpus [12]. Librispeach is recorded with a 16KHz audio frame containing female and male speakers reading segments of book chapters of audio books. It consists of 1000-hour speech data divided into two categories, 'clean' and 'other'. The clean data is automatically filtered and thus considered more easier. 'Other' is anything else, i.e. more challenging data. For each category, there are three splits: dev, test, and train, except for 'other' which has only test and dev sets. Moreover, the clean-train is divided into three subsets, 100, 360, and 500, corresponding to the number of recording hours.

Our experiments were conducted on the train-clean-100 subset, containing 100 recording hours of 251 speakers, as in the CPC paper. We used their same train-test splits and data preprocessing. We first convert the raw files into waveform, and sample segments from the recordings. A segment is defined to be of length 20480 data points, which is around 1.2 seconds. The total number of segments for train-clean-100 is around 30K. However, the encoder is not fed the whole segment, but again a downsampling factor is used. CPC uses a downsampling factor equals 160, making the total number of timesteps T = 20480/160 = 128. To increase the predictive power of the context vector representation, the number of timesteps to be predicted in the future has to increase as well. But that means more computation.

Method	Details	NCE
CPC+GRU	1-layer GRU; 512 hidd.	0.8025
CPC+ATT(1)	4 heads, 512 hidden units	1.1014
CPC+ATT(2)	8 heads, 512 hidden units	1.0912
CPC+ATT(3)	16 heads, 512 hidden units	1.1245

Table 1. NCE loss of different CPC models.

3.2. Implementation Details

For the encoder, we used filter sizes [10, 8, 4, 4, 4] with strides [5, 4, 2, 2, 2] for each convolutional layer, respectively. The number of hidden units used is 512. We used 12 time steps to be predicted in the future to compute the loss. For the GRU, we used 256 hidden units with 1 hidden layer. For the Transformer, we used 512 hidden units, 0.15 drop-out rate, and experimented with different numbers of heads. We used implementation of [10] for the original CPC architecture and training based codes.

3.3. Training

We trained all models for 60 epochs and used Adam optimizer with learning rate scheduler and warm-up steps. We also used Kaiming Initialization to stabilize training.

3.4. Evaluation on Librispeech

We first evaluate the NCE loss of our model and compare its results to the original CPC model, see table 1. The NCE loss of all Transformer-based CPC is close to GRU-based CPC. Model (2), third row, yields the lowest NCE loss within Transformer-based models.

To further test our hypothesis, we use the representations produced by CPC in the unsupervised regime for downstream tasks. We consider the speaker classification task for this purpose. Table 2 shows the result of using a simple classification model on top of the representations from the CPC models. We were able to produce similar results to the original CPC for speaker classification.

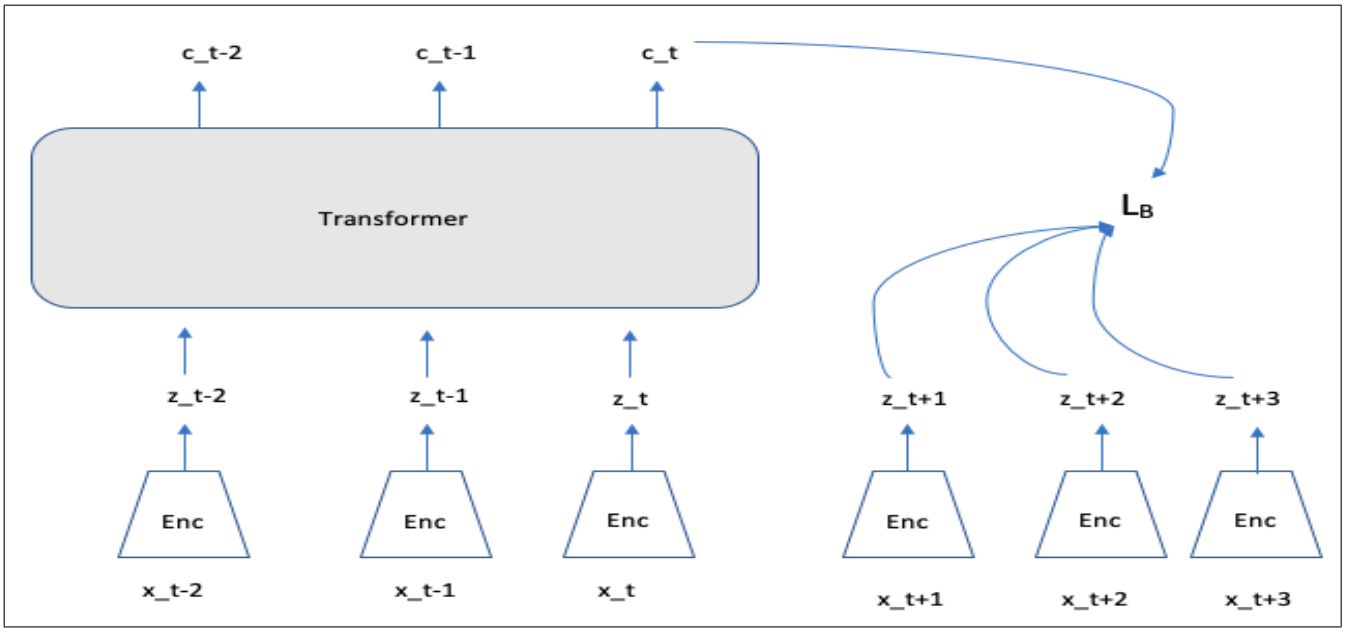

Figure 2. Our approach with Transformer-based CPC.

Method	Speaker Cls. ACC
CPC+GRU	0.94
CPC+ATT(1)	0.92
CPC+ATT(2)	0.91
CPC+ATT(2)	0.94

Table 2. Accuracy achieved by CPC representations. These results are for the validation set, since the models are still running

4. Conclusion and Future Work

In this project, we explore the idea of using Transformers in unsupervised representation learning CPC model, instead of using GRU RNN. We run experiments to test how informative the representations produced by Transformer-based CPC are for downstream tasks, compared to GRU-based CPC. We hope that our work will be a starting point to explore the performance of Transformer-based CPC for other modalities. Other downstream tasks and modalities are left for future work.

Acknowledgements. This work has been done as a course project for the Multimodal Machine Learning class at Boston University. We are deeply thankful for professor Bryan Plummer for his invaluable classes, discussions, and comments.

References

[1] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *CoRR*, abs/2006.11477, 2020.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *CoRR*, abs/2002.05709, 2020.

[4] Ting Chen and Lala Li. Intriguing properties of contrastive losses. *CoRR*, abs/2011.02803, 2020.

[5] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

[6] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. *CoRR*, abs/1505.05192, 2015.

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020.

[8] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceed-*

ings of Machine Learning Research, pages 297–304, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR.

[9] Olivier J. Hénaff, Aravind Srinivas, Jeffrey De Fauw, Ali Razavi, Carl Doersch, S. M. Ali Eslami, and Aäron van den Oord. Data-efficient image recognition with contrastive predictive coding. *CoRR*, abs/1905.09272, 2019.

[10] Cheng-I Lai. Contrastive predictive coding based feature for automatic speaker verification. *arXiv preprint arXiv:1904.01575*, 2019.

[11] Abdelrahman Mohamed, Dmytro Okhonko, and Luke Zettlemoyer. Transformers with convolutional context for ASR. *CoRR*, abs/1904.11660, 2019.

[12] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210, 2015.

[13] Emilio Parisotto, H. Francis Song, Jack W. Rae, Razvan Pascanu, Çaglar Gülçehre, Siddhant M. Jayakumar, Max Jaderberg, Raphael Lopez Kaufman, Aidan Clark, Seb Noury, Matthew M. Botvinick, Nicolas Heess, and Raia Hadsell. Stabilizing transformers for reinforcement learning. *CoRR*, abs/1910.06764, 2019.

[14] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised pre-training for speech recognition. *CoRR*, abs/1904.05862, 2019.

[15] Aravind Srinivas, Michael Laskin, and Pieter Abbeel. CURL: contrastive unsupervised representations for reinforcement learning. *CoRR*, abs/2004.04136, 2020.

[16] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.