

Representation Learning with Contrastive Predictive Coding

Aaron van den Oord
DeepMind
avdnoord@google.com

Yazhe Li
DeepMind
yazhe@google.com

Oriol Vinyals
DeepMind
vinyals@google.com

Abstract

While supervised learning has enabled great progress in many applications, unsupervised learning has not seen such widespread adoption, and remains an important and challenging endeavor for artificial intelligence. In this work, we propose a universal unsupervised learning approach to extract useful representations from high-dimensional data, which we call Contrastive Predictive Coding. The key insight of our model is to learn such representations by predicting the future in latent space by using powerful autoregressive models. We use a probabilistic contrastive loss which induces the latent space to capture information that is maximally useful to predict future samples. It also makes the model tractable by using negative sampling. While most prior work has focused on evaluating representations for a particular modality, we demonstrate that our approach is able to learn useful representations achieving strong performance on four distinct domains: speech, images, text and reinforcement learning in 3D environments.

1 Introduction

Learning high-level representations from labeled data with layered differentiable models in an end-to-end fashion is one of the biggest successes in artificial intelligence so far. These techniques made manually specified features largely redundant and have greatly improved state-of-the-art in several real-world applications [1, 2, 3]. However, many challenges remain, such as data efficiency, robustness or generalization.

Improving representation learning requires features that are less specialized towards solving a single supervised task. For example, when pre-training a model to do image classification, the induced features transfer reasonably well to other image classification domains, but also lack certain information such as color or the ability to count that are irrelevant for classification but relevant for e.g. image captioning [4]. Similarly, features that are useful to transcribe human speech may be less suited for speaker identification, or music genre prediction. Thus, unsupervised learning is an important stepping stone towards robust and generic representation learning.

Despite its importance, unsupervised learning is yet to see a breakthrough similar to supervised learning: modeling high-level representations from raw observations remains elusive. Further, it is not always clear what the ideal representation is and if it is possible that one can learn such a representation without additional supervision or specialization to a particular data modality.

One of the most common strategies for unsupervised learning has been to predict future, missing or contextual information. This idea of predictive coding [5, 6] is one of the oldest techniques in signal processing for data compression. In neuroscience, predictive coding theories suggest that the brain predicts observations at various levels of abstraction [7, 8]. Recent work in unsupervised learning has successfully used these ideas to learn word representations by predicting neighboring words [9]. For images, predicting color from grey-scale or the relative position of image patches has also been

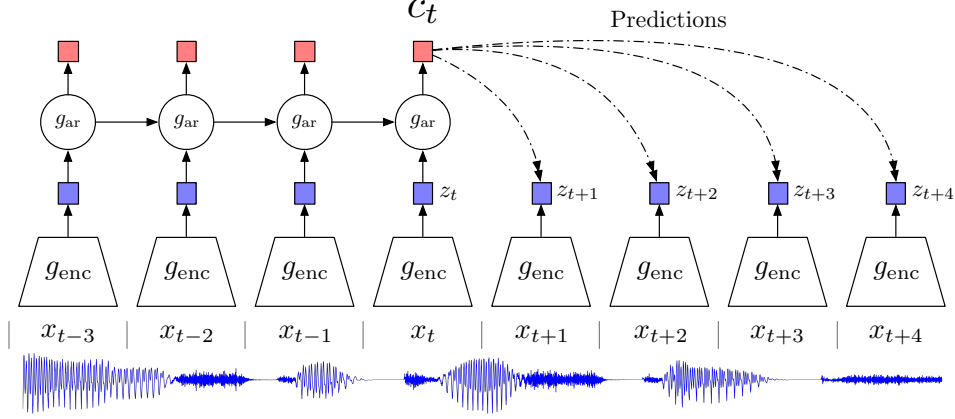


Figure 1: Overview of Contrastive Predictive Coding, the proposed representation learning approach. Although this figure shows audio as input, we use the same setup for images, text and reinforcement learning.

shown useful [10, 11]. We hypothesize that these approaches are fruitful partly because the context from which we predict related values are often conditionally dependent on the same shared high-level latent information. And by casting this as a prediction problem, we automatically infer these features of interest to representation learning.

In this paper we propose the following: first, we compress high-dimensional data into a much more compact latent embedding space in which conditional predictions are easier to model. Secondly, we use powerful autoregressive models in this latent space to make predictions many steps in the future. Finally, we rely on Noise-Contrastive Estimation [12] for the loss function in similar ways that have been used for learning word embeddings in natural language models, allowing for the whole model to be trained end-to-end. We apply the resulting model, Contrastive Predictive Coding (CPC) to widely different data modalities, images, speech, natural language and reinforcement learning, and show that the same mechanism learns interesting high-level information on each of these domains, outperforming other approaches.

2 Contrastive Predicting Coding

We start this section by motivating and giving intuitions behind our approach. Next, we introduce the architecture of **Contrastive Predictive Coding (CPC)**. After that we explain the loss function that is based on Noise-Contrastive Estimation. Lastly, we discuss related work to CPC.

2.1 Motivation and Intuitions

The main intuition behind our model is to learn the representations that encode the underlying shared information between different parts of the (high-dimensional) signal. At the same time it discards low-level information and noise that is more local. In time series and high-dimensional modeling, approaches that use next step prediction exploit the local smoothness of the signal. When predicting further in the future, the amount of shared information becomes much lower, and the model needs to infer more global structure. These 'slow features' [13] that span many time steps are often more interesting (e.g., phonemes and intonation in speech, objects in images, or the story line in books.).

One of the challenges of predicting high-dimensional data is that unimodal losses such as mean-squared error and cross-entropy are not very useful, and powerful conditional generative models which need to reconstruct every detail in the data are usually required. But these models are computationally intense, and waste capacity at modeling the complex relationships in the data x , often ignoring the context c . For example, images may contain thousands of bits of information while the high-level latent variables such as the class label contain much less information (10 bits for 1,024 categories). This suggests that modeling $p(x|c)$ directly may not be optimal for the purpose of extracting shared information between x and c . When predicting future information we instead encode the target x (future) and context c (present) into a compact distributed vector representations (via non-linear

learned mappings) in a way that maximally preserves the mutual information of the original signals x and c defined as

$$I(x; c) = \sum_{x, c} p(x, c) \log \frac{p(x|c)}{p(x)}. \quad (1)$$

By maximizing the mutual information between the encoded representations (which is bounded by the MI between the input signals), we extract the underlying latent variables the inputs have in common.

2.2 Contrastive Predictive Coding

Figure 1 shows the architecture of Contrastive Predictive Coding models. First, a non-linear encoder g_{enc} maps the input sequence of observations x_t to a sequence of latent representations $z_t = g_{\text{enc}}(x_t)$, potentially with a lower temporal resolution. Next, an autoregressive model g_{ar} summarizes all $z_{\leq t}$ in the latent space and produces a **context latent representation** $c_t = g_{\text{ar}}(z_{\leq t})$.

As argued in the previous section we do not predict future observations x_{t+k} directly with a generative model $p_k(x_{t+k}|c_t)$. Instead we model a density ratio which preserves the mutual information between x_{t+k} and c_t (Equation 1) as follows (see next sub-section for further details):

$$f_k(x_{t+k}, c_t) \propto \frac{p(x_{t+k}|c_t)}{p(x_{t+k})} \quad \text{density ratio} \quad (2)$$

where \propto stands for 'proportional to' (i.e. up to a multiplicative constant). Note that the density ratio f can be unnormalized (does not have to integrate to 1). Although any positive real score can be used here, we use a simple log-bilinear model:

$$f_k(x_{t+k}, c_t) = \exp \left(z_{t+k}^T W_k c_t \right), \quad (3)$$

미래의 observation인 x_{t+k} 와 현재의 context 정보인 c_t 사이의 density ratio를 사용하고, 이 density ratio는 이 둘의 mutual information의 양을 보존한다. 즉 mutual information의 proxy로서 이 density ratio를 사용하는 것이다.

In our experiments a linear transformation $W_k^T c_t$ is used for the prediction with a different W_k for every step k . Alternatively, non-linear networks or recurrent neural networks could be used.

By using a density ratio $f(x_{t+k}, c_t)$ and inferring z_{t+k} with an encoder, we relieve the model from modeling the high dimensional distribution x_{t+k} . Although we cannot evaluate $p(x)$ or $p(x|c)$ directly, we can use samples from these distributions, allowing us to use techniques such as **Noise-Contrastive Estimation** [12, 14, 15] and Importance Sampling [16] that are based on comparing the target value with randomly sampled negative values.

In the proposed model, either of z_t and c_t could be used as representation for downstream tasks. The autoregressive model output c_t can be used if extra context from the past is useful. One such example is speech recognition, where the receptive field of z_t might not contain enough information to capture phonetic content. In other cases, where no additional context is required, z_t might instead be better. If the downstream task requires one representation for the whole sequence, as in e.g. image classification, one can pool the representations from either z_t or c_t over all locations.

Finally, note that any type of encoder and autoregressive model can be used in the proposed framework. For simplicity we opted for standard architectures such as strided convolutional layers with resnet blocks for the encoder, and GRUs [17] for the autoregressive model. More recent advancements in autoregressive modeling such as masked convolutional architectures [18, 19] or self-attention networks [20] could help improve results further.

2.3 InfoNCE Loss and Mutual Information Estimation

Both the encoder and autoregressive model are trained to jointly optimize a loss based on NCE, which we will call **InfoNCE**. Given a set $X = \{x_1, \dots, x_N\}$ of N random samples containing one positive sample from $p(x_{t+k}|c_t)$ and $N - 1$ negative samples from the 'proposal' distribution $p(x_{t+k})$, we optimize:

$$\mathcal{L}_N = -\mathbb{E}_X \left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)} \right] \quad (4)$$

Optimizing this loss will result in $f_k(x_{t+k}, c_t)$ estimating the density ratio in equation 2. This can be shown as follows.

The **loss in Equation 4** is the categorical cross-entropy of classifying the positive sample correctly, with $\frac{f_k}{\sum_x f_k}$ being the prediction of the model. Let us write the optimal probability for this loss as $p(d = i|X, c_t)$ with $[d = i]$ being the indicator that sample x_i is the 'positive' sample. The probability that sample x_i was drawn from the conditional distribution $p(x_{t+k}|c_t)$ rather than the proposal distribution $p(x_{t+k})$ can be derived as follows:

$$\begin{aligned} p(d = i|X, c_t) &= \frac{p(x_i|c_t) \prod_{l \neq i} p(x_l)}{\sum_{j=1}^N p(x_j|c_t) \prod_{l \neq j} p(x_l)} \\ &= \frac{\frac{p(x_i|c_t)}{p(x_i)}}{\sum_{j=1}^N \frac{p(x_j|c_t)}{p(x_j)}}. \end{aligned} \quad (5)$$

As we can see, the optimal value for $f(x_{t+k}, c_t)$ in Equation 4 is proportional to $\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}$ and this is independent of the choice of the number of negative samples $N - 1$.

Though not required for training, we can evaluate the mutual information between the variables c_t and x_{t+k} as follows:

$$I(x_{t+k}, c_t) \geq \log(N) - \mathcal{L}_N$$

which becomes tighter as N becomes larger. Also observe that minimizing the **InfoNCE loss \mathcal{L}_N** maximizes a lower bound on mutual information. For more details see Appendix.

2.4 Related Work

CPC is a new method that combines predicting future observations (predictive coding) with a probabilistic contrastive loss (Equation 4). This allows us to extract slow features, which maximize the mutual information of observations over long time horizons. Contrastive losses and predictive coding have individually been used in different ways before, which we will now discuss.

Contrastive loss functions have been used by many authors in the past. For example, the techniques proposed by [21, 22, 23] were based on triplet losses using a max-margin approach to separate positive from negative examples. More recent work includes Time Contrastive Networks [24] which proposes to minimize distances between embeddings from multiple viewpoints of the same scene and whilst maximizing distances between embeddings extracted from different timesteps. In Time Contrastive Learning [25] a contrastive loss is used to predict the segment-ID of multivariate time-series as a way to extract features and perform nonlinear ICA.

There has also been work and progress on defining prediction tasks from related observations as a way to extract useful representations, and many of these have been applied to language. In Word2Vec [9] neighbouring words are predicted using a contrastive loss. Skip-thought vectors [26] and Byte mLSTM [27] are alternatives which go beyond word prediction with a Recurrent Neural Network, and use maximum likelihood over sequences of observations. In Computer Vision [28] use a triplet loss on tracked video patches so that patches from the same object at different timesteps are more similar to each other than to random patches. [11, 29] propose to predict the relative position of patches in an image and in [10] color values are predicted from a greyscale images.

3 Experiments

We present benchmarks on four different application domains: speech, images, natural language and reinforcement learning. For every domain we train CPC models and probe what the representations contain with either a linear classification task or qualitative evaluations, and in reinforcement learning we measure how the auxiliary CPC loss speeds up learning of the agent.

3.1 Audio

For audio, we use a 100-hour subset of the publicly available LibriSpeech dataset [30]. Although the dataset does not provide labels other than the raw text, we obtained force-aligned phone sequences

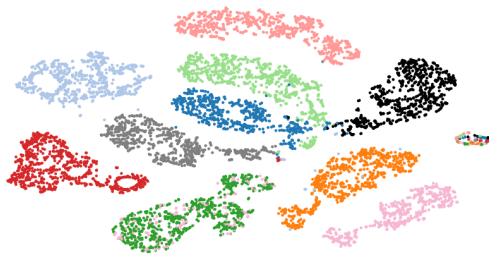


Figure 2: t-SNE visualization of audio (speech) representations for a subset of 10 speakers (out of 251). Every color represents a different speaker.

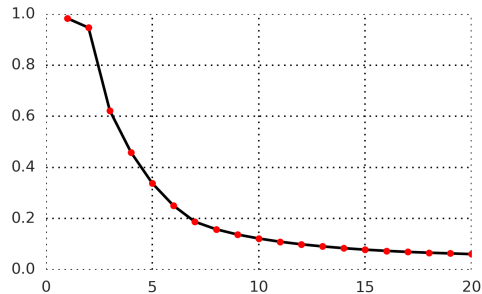


Figure 3: Average accuracy of predicting the positive sample in the contrastive loss for 1 to 20 latent steps in the future of a speech waveform. The model predicts up to 200ms in the future as every step consists of 10ms of audio.

Method	ACC
Phone classification	
Random initialization	27.6
MFCC features	39.7
CPC	64.6
Supervised	74.6
Speaker classification	
Random initialization	1.87
MFCC features	17.6
CPC	97.4
Supervised	98.5

Table 1: LibriSpeech phone and speaker classification results. For phone classification there are 41 possible classes and for speaker classification 251. All models used the same architecture and the same audio input sizes.

Method	ACC
#steps predicted	
2 steps	28.5
4 steps	57.6
8 steps	63.6
12 steps	64.6
16 steps	63.8
Negative samples from	
Mixed speaker	64.6
Same speaker	65.5
Mixed speaker (excl.)	57.3
Same speaker (excl.)	64.6
Current sequence only	65.2

Table 2: LibriSpeech phone classification ablation experiments. More details can be found in Section 3.1.

with the Kaldi toolkit [31] and pre-trained models on Librispeech¹. We have made the aligned phone labels and our train/test split available for download on Google Drive². The dataset contains speech from 251 different speakers.

The encoder architecture g_{enc} used in our experiments consists of a strided convolutional neural network that runs directly on the 16KHz PCM audio waveform. We use five convolutional layers with strides [5, 4, 2, 2, 2], filter-sizes [10, 8, 4, 4, 4] and 512 hidden units with ReLU activations. The total downsampling factor of the network is 160 so that there is a feature vector for every 10ms of speech, which is also the rate of the phoneme sequence labels obtained with Kaldi. We then use a GRU RNN [17] for the autoregressive part of the model, g_{ar} with 256 dimensional hidden state. The output of the GRU at every timestep is used as the context c from which we predict 12 timesteps in the future using the contrastive loss. We train on sampled audio windows of length 20480. We use the Adam optimizer [32] with a learning rate of $2e-4$, and use 8 GPUs each with a minibatch of 8 examples from which the negative samples in the contrastive loss are drawn. The model is trained until convergence, which happens roughly at 300,000 updates.

Figure 3 shows the accuracy of the model to predict latents in the future, from 1 to 20 timesteps. We report the average number of times the logit for the positive sample is higher than for the negative samples in the probabilistic contrastive loss. This figure also shows that the objective is neither trivial nor impossible, and as expected the prediction task becomes harder as the target is further away.

¹www.kaldi-asr.org/downloads/build/6/trunk/egs/librispeech/

²<https://drive.google.com/drive/folders/1BhJ2umKH3whguxMwifaKtSra0TgAbtfb>

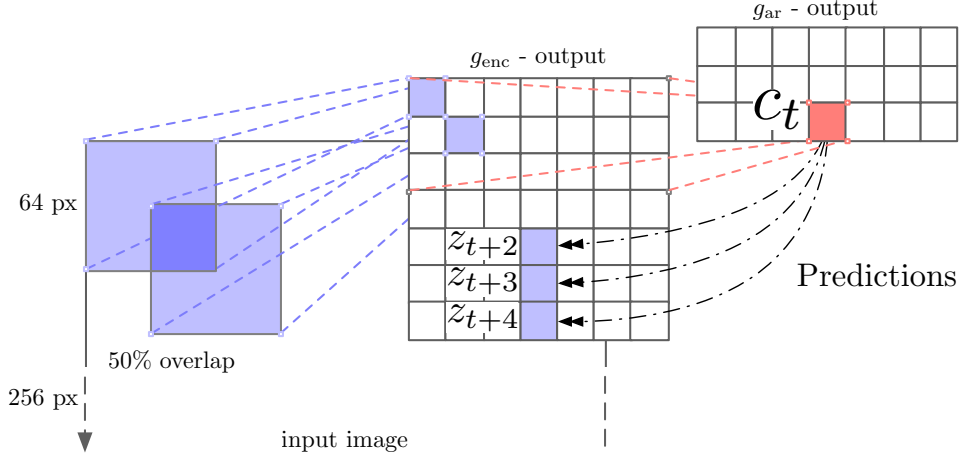


Figure 4: Visualization of Contrastive Predictive Coding for images (2D adaptation of Figure 1).

To understand the representations extracted by CPC, we measure the phone prediction performance with a linear classifier trained on top of these features, which shows how linearly separable the relevant classes are under these features. We extract the outputs of the GRU (256 dimensional), i.e. c_t , for the whole dataset after model convergence and train a multi-class linear logistic regression classifier. The results are shown in Table 1 (top). We compare the accuracy with three baselines: representations from a random initialized model (i.e., g_{enc} and g_{ar} are untrained), MFCC features, and a model that is trained end-to-end supervised with the labeled data. These two models have the same architecture as the one used to extract the CPC representations. The fully supervised model serves as an indication for what is achievable with this architecture. We also found that not all the information encoded is linearly accessible. When we used a single hidden layer instead the accuracy increases from **64.6** to **72.5**, which is closer to the accuracy of the fully supervised model.

Table 2 gives an overview of two ablation studies of CPC for phone classification. In the first set we vary the number of steps the model predicts showing that predicting multiple steps is important for learning useful features. In the second set we compare different strategies for drawing negative sample, all predicting 12 steps (which gave the best result in the first ablation). In the mixed speaker experiment the negative samples contain examples of different speakers (first row), in contrast to same speaker experiment (second row). In the third and fourth experiment we exclude the current sequence to draw negative samples from (so only other examples in the minibatch are present in X) and in the last experiment we only draw negative samples within the sequence (thus all samples are from the same speaker).

Beyond phone classification, Table 1 (bottom) shows the accuracy of performing speaker identity (out of 251) with a linear classifier from the same representation (we do not average utterances over time). Interestingly, CPCs capture both speaker identity and speech contents, as demonstrated by the good accuracies attained with a simple linear classifier, which also gets close to the oracle, fully supervised networks.

Additionally, Figure 2 shows a t-SNE visualization [33] of how discriminative the embeddings are for speaker voice-characteristics. It is important to note that the window size (maximum context size for the GRU) has a big impact on the performance, and longer segments would give better results. Our model had a maximum of 20480 timesteps to process, which is slightly longer than a second.

3.2 Vision

In our visual representation experiments we use the ILSVRC ImageNet competition dataset [34]. The ImageNet dataset has been used to evaluate unsupervised vision models by many authors [28, 11, 35, 10, 29, 36]. We follow the same setup as [36] and use a ResNet v2 101 architecture [37] as the image encoder g_{enc} to extract CPC representations (note that this encoder is not pretrained). We did not use Batch-Norm [38]. After unsupervised training, a linear layer is trained to measure classification accuracy on ImageNet labels.

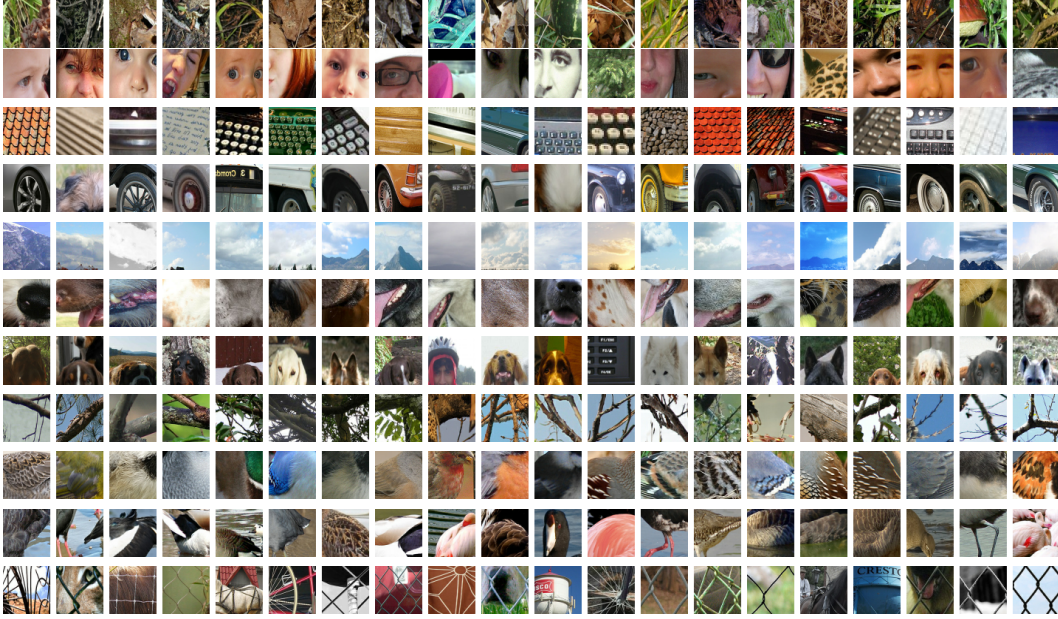


Figure 5: Every row shows image patches that activate a certain neuron in the CPC architecture.

The training procedure is as follows: from a 256x256 image we extract a 7x7 grid of 64x64 crops with 32 pixels overlap. Simple data augmentation proved helpful on both the 256x256 images and the 64x64 crops. The 256x256 images are randomly cropped from a 300x300 image, horizontally flipped with a probability of 50% and converted to greyscale. For each of the 64x64 crops we randomly take a 60x60 subcrop and pad them back to a 64x64 image.

Each crop is then encoded by the ResNet-v2-101 encoder. We use the outputs from the third residual block, and spatially mean-pool to get a single 1024-d vector per 64x64 patch. This results in a 7x7x1024 tensor. Next, we use a PixelCNN-style autoregressive model [19] (a convolutional row-GRU PixelRNN [39] gave similar results) to make predictions about the latent activations in following rows top-to-bottom, visualized in Figure 4. We predict up to five rows from the 7x7 grid, and we apply the contrastive loss for each patch in the row. We used Adam optimizer with a learning rate of $2e-4$ and trained on 32 GPUs each with a batch size of 16.

For the linear classifier trained on top of the CPC features we use SGD with a momentum of 0.9, a learning rate schedule of 0.1, 0.01 and 0.001 for 50k, 25k and 10k updates and batch size of 2048 on a single GPU. Note that when training the linear classifier we first spatially mean-pool the 7x7x1024 representation to a single 1024 dimensional vector. This is slightly different from [36] which uses a 3x3x1024 representation without pooling, and thus has more parameters in the supervised linear mapping (which could be advantageous).

Tables 3 and 4 show the top-1 and top-5 classification accuracies compared with the state-of-the-art. Despite being relatively domain agnostic, CPCs improve upon state-of-the-art by 9% absolute in top-1 accuracy, and 4% absolute in top-5 accuracy.

3.3 Natural Language

Our natural language experiments follow closely the procedure from [26] which was used for the skip-thought vectors model. We first learn our unsupervised model on the BookCorpus dataset [42], and evaluate the capability of our model as a generic feature extractor by using CPC representations for a set of classification tasks. To cope with words that are not seen during training, we employ vocabulary expansion the same way as [26], where a linear mapping is constructed between word2vec and the word embeddings learned by the model.

For the classification tasks we used the following datasets: movie review sentiment (MR) [43], customer product reviews (CR) [44], subjectivity/objectivity [45], opinion polarity (MPQA) [46] and question-type classification (TREC) [47]. As in [26] we train a logistic regression classifier and

Method	Top-1 ACC
Using AlexNet conv5	
Video [28]	29.8
Relative Position [11]	30.4
BiGan [35]	34.8
Colorization [10]	35.2
Jigsaw [29] *	38.1
Using ResNet-V2	
Motion Segmentation [36]	27.6
Exemplar [36]	31.5
Relative Position [36]	36.2
Colorization [36]	39.6
CPC	48.7

Table 3: ImageNet top-1 unsupervised classification results. *Jigsaw is not directly comparable to the other AlexNet results because of architectural differences.

Method	Top-5 ACC
Motion Segmentation (MS)	48.3
Exemplar (Ex)	53.1
Relative Position (RP)	59.2
Colorization (Col)	62.5
Combination of MS + Ex + RP + Col	69.3
CPC	73.6

Table 4: ImageNet top-5 unsupervised classification results. Previous results with MS, Ex, RP and Col were taken from [36] and are the best reported results on this task.

Method	MR	CR	Subj	MPQA	TREC
Paragraph-vector [40]	74.8	78.1	90.5	74.2	91.8
Skip-thought vector [26]	75.5	79.3	92.1	86.9	91.4
Skip-thought + LN [41]	79.5	82.6	93.4	89.0	-
CPC	76.9	80.1	91.2	87.7	96.8

Table 5: Classification accuracy on five common NLP benchmarks. We follow the same transfer learning setup from Skip-thought vectors [26] and use the BookCorpus dataset as source. [40] is an unsupervised approach to learning sentence-level representations. [26] is an alternative unsupervised learning approach. [41] is the same skip-thought model with layer normalization trained for 1M iterations.

evaluate with 10-fold cross-validation for MR, CR, Subj, MPQA and use the train/test split for TREC. A L2 regularization weight was chosen via cross-validation (therefore nested cross-validation for the first 4 datasets).

Our model consists of a simple sentence encoder g_{enc} (a 1D-convolution + ReLU + mean-pooling) that embeds a whole sentence into a 2400-dimension vector z , followed by a GRU (2400 hidden units) which predicts up to 3 future sentence embeddings with the contrastive loss to form c . We used Adam optimizer with a learning rate of $2e-4$ trained on 8 GPUs, each with a batch size of 64. We found that more advanced sentence encoders did not significantly improve the results, which may be due to the simplicity of the transfer tasks (e.g., in MPQA most datapoints consists of one or a few words), and the fact that bag-of-words models usually perform well on many NLP tasks [48].

Results on evaluation tasks are shown in Table 5 where we compare our model against other models that have been used using the same datasets. The performance of our method is very similar to the skip-thought vector model, with the advantage that it does not require a powerful LSTM as word-level decoder, therefore much faster to train. Although this is a standard transfer learning benchmark, we found that models that learn better relationships in the children books did not necessarily perform better on the target tasks (which are very different: movie reviews etc). We note that better [49, 27] results have been published on these target datasets, by transfer learning from a different source task.

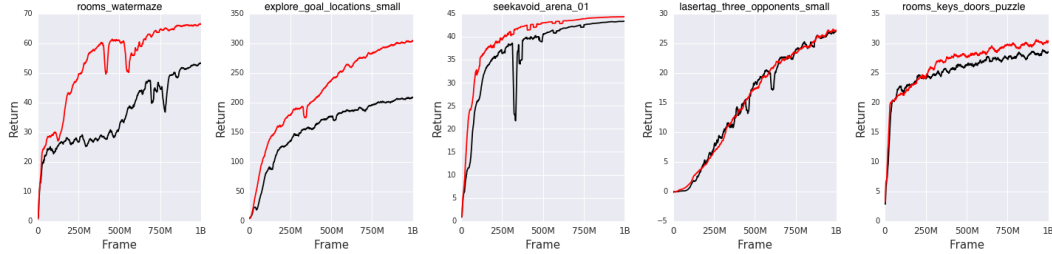


Figure 6: Reinforcement Learning results for 5 DeepMind Lab tasks used in [50]. Black: batched A2C baseline, Red: with auxiliary contrastive loss.

3.4 Reinforcement Learning

Finally, we evaluate the proposed unsupervised learning approach on five reinforcement learning in 3D environments of DeepMind Lab [51]: `rooms_watermaze`, `explore_goal_locations_small`, `seekavoid_arena_01`, `lasertag_three_opponents_small` and `rooms_keys_doors_puzzle`.

This setup differs from the previous three. Here, we take the standard batched A2C [52] agent as base model and add CPC as an auxiliary loss. We do not use a replay buffer, so the predictions have to adapt to the changing behavior of the policy. The learned representation encodes a distribution over its future observations.

Following the same approach as [50], we perform a random search over the entropy regularization weight, the learning-rate and epsilon hyperparameters for RMSProp [53]. The unroll length for the A2C is 100 steps and we predict up to 30 steps in the future to derive the contrastive loss. The baseline agent consists of a convolutional encoder which maps every input frame into a single vector followed by a temporal LSTM. We use the same encoder as in the baseline agent and only add the linear prediction mappings for the contrastive loss, resulting in minimal overhead which also showcases the simplicity of implementing our method on top of an existing architecture that has been designed and tuned for a particular task. We refer to [50] for all other hyperparameter and implementation details.

Figure 6 shows that for 4 out of the 5 games performance of the agent improves significantly with the contrastive loss after training on 1 billion frames. For `lasertag_three_opponents_small`, contrastive loss does not help nor hurt. We suspect that this is due to the task design, which does not require memory and thus yields a purely reactive policy.

4 Conclusion

In this paper we presented **Contrastive Predictive Coding (CPC)**, a framework for extracting compact latent representations to encode predictions over future observations. CPC combines autoregressive modeling and noise-contrastive estimation with intuitions from predictive coding to learn abstract representations in an unsupervised fashion. We tested these representations in a wide variety of domains: audio, images, natural language and reinforcement learning and achieve strong or state-of-the-art performance when used as stand-alone features. The simplicity and low computational requirements to train the model, together with the encouraging results in challenging reinforcement learning domains when used in conjunction with the main loss are exciting developments towards useful unsupervised learning that applies universally to many more data modalities.

5 Acknowledgements

We would like to thank Andriy Mnih, Andrew Zisserman, Alex Graves and Carl Doersch for their helpful comments on the paper and Lasse Espeholt for making the A2C baseline available.

References

- [1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

- [2] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.
- [3] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.
- [4] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 3156–3164. IEEE, 2015.
- [5] Peter Elias. Predictive coding–i. *IRE Transactions on Information Theory*, 1(1):16–24, 1955.
- [6] Bishnu S Atal and Manfred R Schroeder. Adaptive predictive coding of speech signals. *The Bell System Technical Journal*, 49(8):1973–1986, 1970.
- [7] Rajesh PN Rao and Dana H Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience*, 2(1):79, 1999.
- [8] Karl Friston. A theory of cortical responses. *Philosophical transactions of the Royal Society B: Biological sciences*, 360(1456):815–836, 2005.
- [9] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
- [10] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European Conference on Computer Vision*, pages 649–666. Springer, 2016.
- [11] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- [12] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304, 2010.
- [13] Laurenz Wiskott and Terrence J Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural computation*, 14(4):715–770, 2002.
- [14] Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic language models. *arXiv preprint arXiv:1206.6426*, 2012.
- [15] Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*, 2016.
- [16] Yoshua Bengio and Jean-Sebastien Senecal. Adaptive importance sampling to accelerate training of a neural probabilistic language model. *IEEE Trans. Neural Networks*, 19, 2008.
- [17] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.
- [18] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.
- [19] Aaron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. Conditional image generation with pixelcnn decoders. *CoRR*, abs/1606.05328, 2016.
- [20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017.
- [21] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 539–546. IEEE, 2005.
- [22] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(Feb):207–244, 2009.

- [23] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.
- [24] Pierre Sermanet, Corey Lynch, Jasmine Hsu, and Sergey Levine. Time-contrastive networks: Self-supervised learning from multi-view observation. *arXiv preprint arXiv:1704.06888*, 2017.
- [25] Aapo Hyvarinen and Hiroshi Morioka. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3765–3773. Curran Associates, Inc., 2016.
- [26] Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302, 2015.
- [27] Alec Radford, Rafal Jozefowicz, and Ilya Sutskever. Learning to generate reviews and discovering sentiment. *arXiv preprint arXiv:1704.01444*, 2017.
- [28] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. *arXiv preprint arXiv:1505.00687*, 2015.
- [29] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016.
- [30] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 5206–5210. IEEE, 2015.
- [31] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011.
- [32] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [33] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.
- [34] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.
- [35] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.
- [36] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2051–2060, 2017.
- [37] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645. Springer, 2016.
- [38] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456, 2015.
- [39] Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *CoRR*, abs/1601.06759, 2016.
- [40] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196, 2014.
- [41] Lei Jimmy Ba, Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016.
- [42] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015.

- [43] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 115–124. Association for Computational Linguistics, 2005.
- [44] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM, 2004.
- [45] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, page 271. Association for Computational Linguistics, 2004.
- [46] Janyce Wiebe, Theresa Wilson, and Claire Cardie. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3):165–210, 2005.
- [47] Xin Li and Dan Roth. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics, 2002.
- [48] Sida Wang and Christopher D. Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL ’12, pages 90–94, 2012.
- [49] Han Zhao, Zhengdong Lu, and Pascal Poupart. Self-adaptive hierarchical sentence model. In *IJCAI*, pages 4069–4076, 2015.
- [50] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymyr Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. *arXiv preprint arXiv:1802.01561*, 2018.
- [51] Charles Beattie, Joel Z Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, et al. Deepmind lab. *arXiv preprint arXiv:1612.03801*, 2016.
- [52] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937, 2016.
- [53] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning-lecture 6a-overview of mini-batch gradient descent.
- [54] Ishmael Belghazi, Sai Rajeswar, Aristide Baratin, R Devon Hjelm, and Aaron Courville. Mine: Mutual information neural estimation. 2018.

A Appendix

A.1 Estimating the Mutual Information with InfoNCE

By optimizing InfoNCE, the CPC loss we defined in Equation 4, we are maximizing the mutual information between c_t and z_{t+k} (which is bounded by the MI between c_t and x_{t+k}). This can be shown as follows.

As already shown in Section 2.3, the optimal value for $f(x_{t+k}, c_t)$ is given by $\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}$. Inserting this back in to Equation 4 and splitting X into the positive example and the negative examples X_{neg} results in:

$$\mathcal{L}_N^{\text{opt}} = -\mathbb{E}_X \log \left[\frac{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}}{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})} + \sum_{x_j \in X_{\text{neg}}} \frac{p(x_j|c_t)}{p(x_j)}} \right] \quad (6)$$

$$= \mathbb{E}_X \log \left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} \sum_{x_j \in X_{\text{neg}}} \frac{p(x_j|c_t)}{p(x_j)} \right] \quad (7)$$

$$\approx \mathbb{E}_X \log \left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} (N-1) \mathbb{E}_{x_j} \frac{p(x_j|c_t)}{p(x_j)} \right] \quad (8)$$

$$= \mathbb{E}_X \log \left[1 + \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} (N-1) \right] \quad (9)$$

$$\geq \mathbb{E}_X \log \left[\frac{p(x_{t+k})}{p(x_{t+k}|c_t)} N \right] \quad (10)$$

$$= -I(x_{t+k}, c_t) + \log(N), \quad (11)$$

Therefore, $I(x_{t+k}, c_t) \geq \log(N) - \mathcal{L}_N^{\text{opt}}$. This trivially also holds for other f that obtain a worse (higher) \mathcal{L}_N . Equation 8 quickly becomes more accurate as N increases. At the same time $\log(N) - \mathcal{L}_N$ also increases, so it's useful to use large values of N .

InfoNCE is also related to MINE [54]. Without loss of generality let's write $f(x, c) = e^{F(x, c)}$, then

$$\mathbb{E}_X \left[\log \frac{f(x, c)}{\sum_{x_j \in X} f(x_j, c)} \right] = \mathbb{E}_{(x, c)} [F(x, c)] - \mathbb{E}_{(x, c)} \left[\log \sum_{x_j \in X} e^{F(x_j, c)} \right] \quad (12)$$

$$= \mathbb{E}_{(x, c)} [F(x, c)] - \mathbb{E}_{(x, c)} \left[\log \left(e^{F(x, c)} + \sum_{x_j \in X_{\text{neg}}} e^{F(x_j, c)} \right) \right] \quad (13)$$

$$\leq \mathbb{E}_{(x, c)} [F(x, c)] - \mathbb{E}_c \left[\log \sum_{x_j \in X_{\text{neg}}} e^{F(x_j, c)} \right] \quad (14)$$

$$= \mathbb{E}_{(x, c)} [F(x, c)] - \mathbb{E}_c \left[\log \frac{1}{N-1} \sum_{x_j \in X_{\text{neg}}} e^{F(x_j, c)} + \log(N-1) \right] \quad (15)$$

is equivalent to the MINE estimator (up to a constant). So we maximize a lower bound on this estimator. We found that using MINE directly gave identical performance when the task was non-trivial, but became very unstable if the target was easy to predict from the context (e.g., when predicting a single step in the future and the target overlaps with the context).