Representation learning with contrastive predictive coding(CPC)

`20221128, Mon.`

참고 논문 : 1807.03748.pdf (arxiv.org)

▼ ref

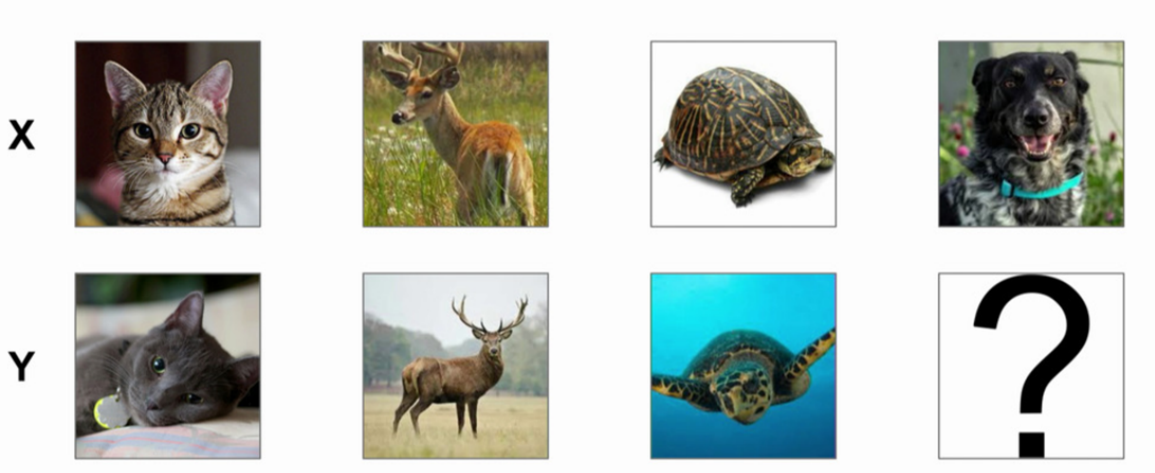
Contrastive Predictive Coding (CPC) 리뷰 :: Kaen's Ritus (tistory.com)

Data-Efficient Image Recognition With Contrastive Predictive Coding - Paper Note (creamnuts.github.io)

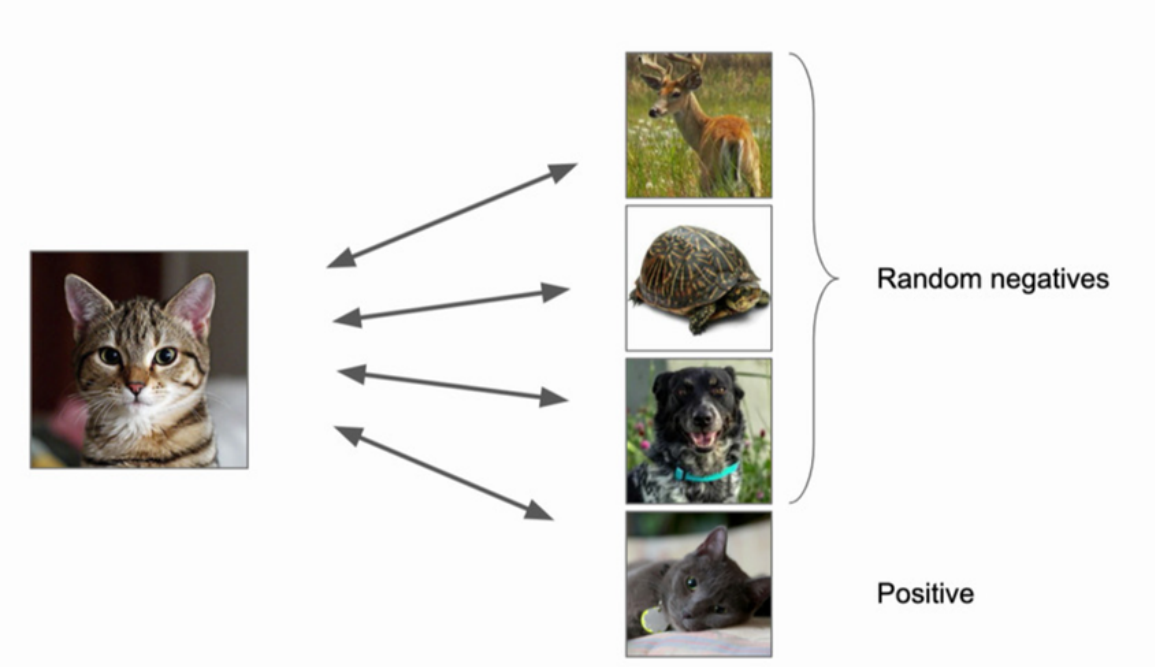
From the bottom :: Representation Learning with Contrastive Predictive Coding (CPC) (tistory.com)

Overview

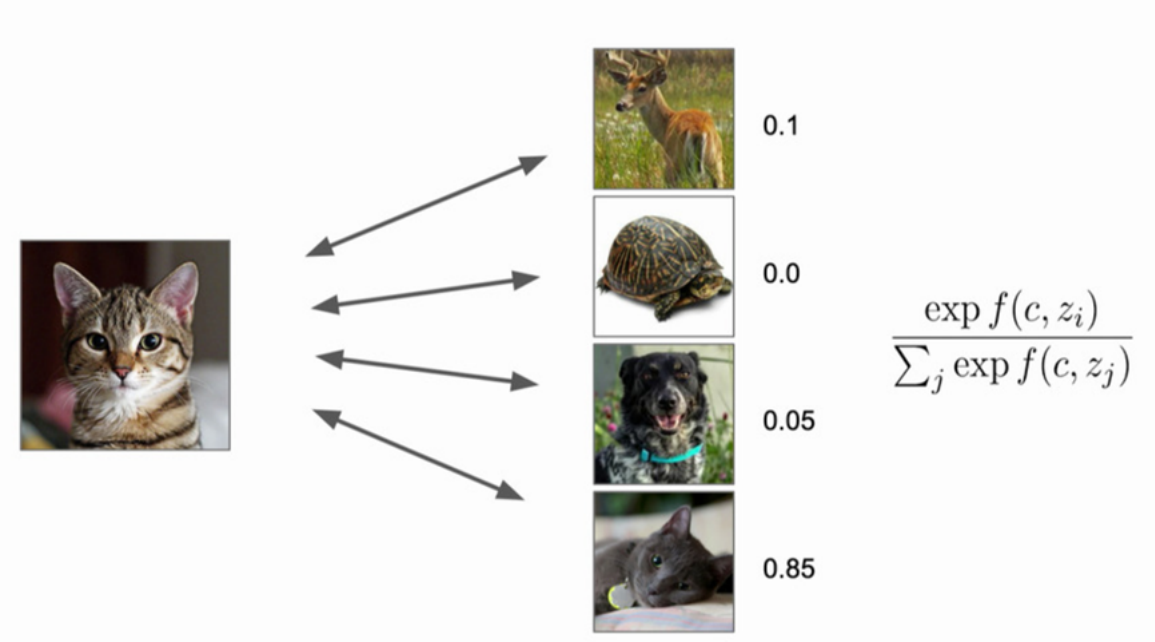
- 이 논문은 high-dimensional 데이터에서 유용한 representation을 추출하기 위한 unsupervised learning 접근법을 제안
- 강력한 auto-regressive 모델을 사용하여 latent space 에서 추출된 z를 이용하여 미래를 예측하는 representation 학습법 제안
- 미래 샘플을 예측할 때 가장 유용한 정보를 얻기 위해 latent space를 유도하는 확률적 contrastive loss 사용
- Mutual information이란? [1]



Mutual information



Mutual information - Contrastive



Mutual information - Contrastive loss

Background

- 지도 학습 기반의 end-to-end 네트워크는 AI를 발전시키는 것에 성공했지만, 데이터 효율성, 견고성 또는 일반화 같은 많은 과제가 남아 있음
- 이를 위해 high-level representation 학습이 가능한 unsupervised 기법이 필요하지만, raw level observation에서 high-level representation 모델링이 어렵고, 이상적인 표현이 무엇인지 명확하지 않으며, 특정 데이터 양식에 대한 추가 감독이나 전문화 없이 이러한 표현을 배울 수 있는지 명확하지 않음
- unsupervised 학습을 위한 가장 일반적인 전략 중 하나는 미래를 예측하는 것이며, predict coding 아이디어는 데이터 압축을 위한 신호 처리에서 사용된 오래된 기술 중 하나임

- 또한, 미래나 contextual 정보를 예측하는 unsupervised learning 기법을 제안하고자 함
- 실제로 이러한 아이디어를 사용하여 이웃 단어를 예측하여 단어 표현을 학습한 사례가 있음 (Word2Vec) [2]

Proposed method

1. conditional 예측 모델링을 쉽게 하기 위해 latent embedding space로 high-dimensional data를 압축
   - 이 latent space에서 강력한 autoregressive model을 사용하여 미래에 많은 step들을 예측함
   - 즉, 고차원 데이터 간에 shared information을 인코딩한 mutual information을 얻기 위해 representation을 학습함
2. Noise-Contrastive Estimation[3]에 의존한 loss 함수
   - 자연어 모델에서 word embedding을 학습하는데 사용된 것과 유사한 방식으로, loss 함수에 대해 Noise-Contrastive Estimation에 의존하여 전체 모델이 end-to-end 학습 가능하도록 제안
3. 결과 모델인 Contrastive Predictive Coding (CPC)를 다양한 task에 적용
   - 이미지, 음성, 자연어 및 강화 학습

Noise-Contrastive Estimation

1. NCE
   a. CBOW와 Skip-Gram 모델에서 사용하는 비용 계산 알고리즘
      - 전체 데이터셋에 대해 SoftMax 함수를 적용하는 것이 아니라 샘플링으로 추출한 일부에만 적용하는 방법
      - k개의 대비되는(contrastive) 단어들을 noise distribution으로부터 구해서 (몬테카를로) 평균을 구하는 것이 기본 알고리즘 : Hierarchical SoftMax와 Negative Sampling 등 여러 가지 방법 존재
      - 일반적으로 단어 개수가 많을 때 사용하고, NCE를 사용하면 문제를 (실제 분포에서 얻은 샘플=Positive)과 (인공적으로 만든 잡음 분포에서 얻은 샘플=Negative)을 구별하는 이진 분류 문제로 바꿀 수 있게 됨
   b. Negative Sampling에서 사용하는 목적 함수는 결과값이 최대화될 수 있는 형태로 구성
      - 현재(목표, target, positive) 단어에는 높은 확률을 부여하고, 나머지 단어(negative, noise)에는 낮은 확률을 부여해서 가장 큰 값을 만들 수 있는 공식 사용
      - 특히, 계산 비용에서 전체 단어를 계산하는 것이 아니라 선택한 k개의 noise 단어들만 계산하면 되기 때문에 효율적
2. Hierarchical SoftMax
   a. CBOW와 Skip-Gram 모델은 내부적으로 SoftMax 알고리즘을 사용해서 계산 진행함
      - 모든 단어에 대해 계산 및 normalization을 진행하는데 많은 시간이 걸리는 단점
      - 이를 해결하기 위한 Hierarchical SoftMax와 Negative Sampling 알고리즘이 있음
   b. Hierarchical SoftMax
      - Hierarchical SoftMax 알고리즘은 계산량이 많은 SoftMax function을 빠르게 계산 가능한 multinomial distribution 함수로 대체함
      - Word2Vec 논문에서는 사용 빈도가 높은 단어에 대해 짧은 경로를 부여함
3. Negative Sampling

a. SoftMax 알고리즘을 몇 개의 샘플에만 적용하는 알고리즘
   - 전체 데이터로부터 일부만 뽑아서 SoftMax 계산을 수행하고 normalization을 진행
   - 이때 현재(목표) 단어는 반드시 계산을 수행해야 하기 때문에 Positive Sample이라 부름
   - 나머지 단어를 Negative Sample이라 부름

b. Negative Sampling에서는 나머지 단어에 해당하는 Negative Sample 추출 방법이 핵심
   - 일반적으로 샘플링은 'Noise Distribution'을 정의하고 그 분포를 이용하여 일정 개수를 추출함

즉 정리해보면,

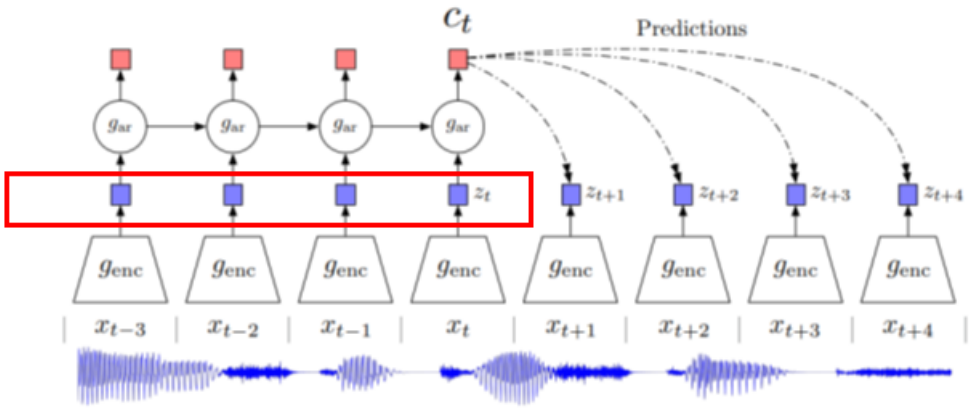
CPC = 미래 관찰 예측(예측 코딩) + 확률적 대비 손실의 결합으로 볼 수 있음

이를 통해,

- 오랜 시간 동안 data observation의 mutual information을 최대화 할 수 있음
- 여러 시간 간격으로 떨어져 있는 데이터 포인트들에서 shared information을 인코딩하여 representation 학습
- 이러한 feature를 'Slow Features'라고 부르며, 이는 시간이 빠르게 지나도 변하지 않는 feature를 의미함
- E.g., 오디오 신호에서 말하는 사람, 비디오 내의 흐름 등

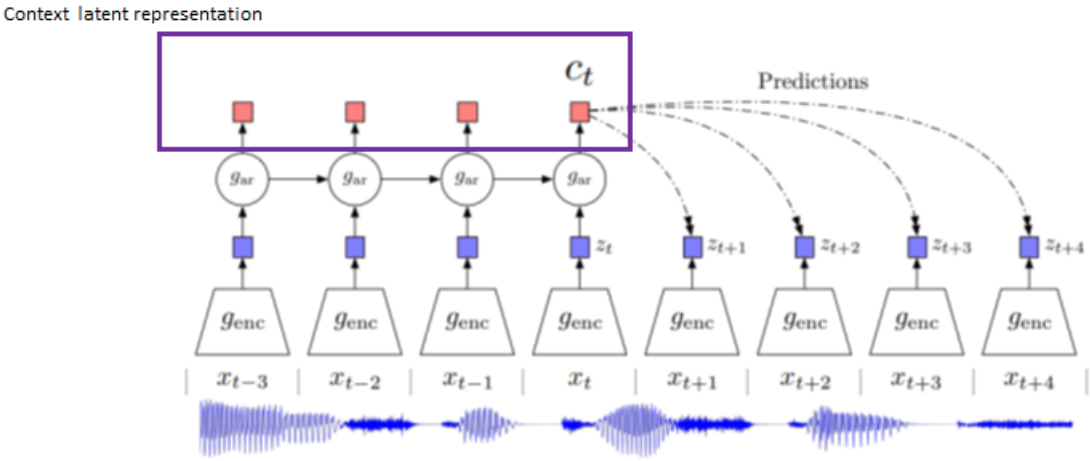
Model architecture

1. 비선형 인코더, g_{enc}
   - 해당 time step t를 기준으로 진행되며, 이후의 값들을 사용하여 예측하지 않음
   - 관측치 x_t인 input sequence를 $z_t = g_{enc}(x_t)$ 로 매핑함
   - 이를 통해, 잠재적으로 더 lower한 temporal resolution으로 변경됨



CPC Encoder

2. Autoregressive model, g_{ar}
   - Latent space의 현재 시간 t를 포함한 이전의 모든 값들인 $z \leq t$ 에 대해 요약
   - 이후 Context latent representation인 $c_t = g_{ar}(z \leq t)$ 를 생성



CPC context nextwork

- 이때, 생성 모델 $p_k(x_{t+k}|c_t)$ 를 사용하여 미래 관측치인 x_{t+k} 를 직접 예측하지 않음
- 대신 다음과 같이 x_{t+k} 와 c_t 사이의 mutual information을 보존하는 식(1)을 활용하여 density 비율을 모델링 함

$$I(x;c) = \sum_{x,c} p(x,c) \log \frac{p(x_{t+k}|c_t)}{p(x_{t+k})} \quad \cdots (1)$$

$$f_k(x_{t_k}, c_t) \propto \frac{p(x_{t+k}|c_t)}{p(x_{t+k})} \quad \cdots (2)$$

- 식(2)인 밀도 비율 f 는 정규화 되지 않을 수 있으므로, log-bilinear model을 적용함

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

- 해당 논문에서, linear transformation인 $W_k c_t$ 는 모든 k-단계에 대해 다른 W_k 를 갖는 예측에 사용됨
- 혹은 비선형 네트워크 또는 반복 신경망으로 사용 가능
- Density ratio $f(x_{t+k}, c_t)$ 와 인코더로 z_{t_k} 를 추론함으로써 모델이 고차원 분포인 x_{t_k} 를 모델링 하지 않도록 함
- 이 때, $p(x)$혹은 $p(x|c)$를 직접 평가할 수는 없지만, 위의 분포를 Noise-Contrastive Estimation 혹은 Importance Sampling 방법으로 평가함
- 목표인 Positive 값과 무작위로 샘플링 된 Negative 값의 비교를 기반으로 평가함
- 이를 위해 InfoNCE loss 사용!

3. Downstream task
   a. z_t 및 c_t중 하나를 downstream 작업에 대한 representation으로 사용
      - Autoregressive model 출력인 c_t는 과거의 extra context가 유용할 경우 사용 (음성 인식, phoneme, speech 등)
      - Extra context가 필요하지 않은 다른 경우, z_t를 사용하는 것이 유용 (비전, 이미지 등)
   b. 논문에서는 단순화를 위해,
      - Encoder : Strided CNN with ResNet blocks
      - Decoder : GRU

을 사용하였고, Masking CNN, Self-attention 등 사용하는 것을 권장

c. **InfoNCE** loss

- Encoder와 Autoregressive model 모두 NCE 기반의 Loss로 최적화 되며, 이를 논문에서 InfoNCE라 부름
- $p(x_{t+k}|c_t)$의 Positive sample 하나와 $proposal$분포 $p(x_{t+k})$의 $N-1$개의 Negative 샘플을 포함하는 N개의 random sample 집합 $X = x_1, x_2, ..., x_N$이 주어지면, 아래의 식을 최적화함

$$\mathcal{L}_N = E_X[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}] \quad \cdots (4)$$

- 위의 식 loss를 최적화하면 $f_k(x_{t+k}, c_t)$가 식 (2)의 density 비율을 추정함
- 식 (4)의 loss는 Positive 샘플을 바르게 분류하는 categorical cross-entropy이며, $\frac{f_k}{\sum_X f_k}$는 모델의 예측임
- 위 loss에 대한 최적의 확률을 $p(d = i|X, c_t)$라 할때, $[d = i]$는 샘플 x_i의 Positive 샘플링 됨
- 제안된 분포 $p(x_{t+k})$가 아닌 조건부 분포 $p(x_{t+k}|c_k)$에서 샘플 x_i를 추출할 확률은 다음과 같음

$$p(d = i|X, c_t) = \frac{p(x_i|c_t)\Pi_{l \neq j} p(x_l)}{\sum_{j=1}^{N} p(x_j|c_t)\Pi_{l \neq j} p(x_l)} \quad \cdots (5)$$

$$= \frac{\frac{p(x_i|c_t)}{p(x_i)}}{\sum_{j=1}^{N} \frac{p(x_j|c_t)}{x_j}} \quad \cdots (6)$$

- 식 (4)에서 $f(x_{t+k}, c_t)$에 대한 최적값은 $\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}$에 비례하며, 이는 Negative 샘플 수인 $N-1$과 무관함

Experiments (여기에선 Phoneme recognition에 관해서만 다루겠음)

1. Using LibriSpeech-100hour train
   - Kaldi toolkit 및 LibriSpeech1에서 사전 훈련된 모델을 사용하여 강제 정렬된 시퀀스 사용
   - 251명의 서로 다른 화자의 speech 포함
2. Encoder
   - 입력 음성 16kHz PCM 그대로 입력 받음
   - Strides [5,4,2,2,2], filter-sizes [10,8,4,4,4], 512 차원의 ReLU사용
   - Downsampling factor가 160이므로, 음성의 10ms마다 특성 벡터가 사용됨
3. Autoregressive
   - 256차원의 GRU RNN 사용
   - 모든 시간 단계에서 GRU의 출력은 contrastive loss를 사용하여 12개의 시간 단계를 예측하는 컨텍스트 c로 사용됨
   - 음성 길이는 20480으로 fix
4. Hyperparameters
   - Adam optimizer, 2e-4 lr

- Contrastive loss의 negative sample을 추출하는 Minibatch size 8로 GPU 8개 학습
- 수렴될 때까지 학습, 대략 300,000 steps 학습

Results

1. Predict latents in the future (Figure 3)

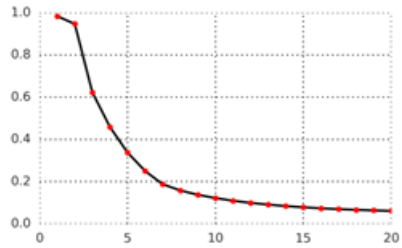


Figure 3: Average accuracy of predicting the positive sample in the contrastive loss for 1 to 20 latent steps in the future of a speech waveform. The model predicts up to 200ms in the future as every step consists of 10ms of audio.

미래 예측 step에 따른 정확도

- Time step으로 미래의 latent를 예측하는 모델의 정확도
- Positive sample에 대한 logit이 Negative sample 보다 더 높은 평균 횟수를 가짐
- 거리가 멀어질수록 waveform 예측 정확도가 떨어짐

2. Phoneme classification (Table 1)

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.

Phoneme recognition 결과

- 모델 수렴 후 전체 데이터 세트에 대한 GRU (256차원), 즉 c_t의 출력을 추출
- Multi-class linear logistic regression classifier 훈련 (downstream)
- MFCC, CPC, Supervised 모두 같은 네트워크 구조
- GRU 대신 single hidden layer 사용하였을 경우 72.5%

3. Two ablation studies of CPC (Table 2)

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.

미래 예측 step에 따른 정확도

- Time step의 거리가 멀어질수록 waveform 예측 정확도가 낮지만, 예측하는 미래 step 수가 멀어질수록 음소 분류 정확도가 높아짐
- 모든 샘플이 동일한 스피커로부터 얻어졌을 경우, 음소 분류 정확도가 높음을 알 수 있음

4. Speaker identification
   - 동일한 representation의 linear classifier를 사용하여 251명의 speaker identity 수행
   - 단순한 linear classifier로 얻은 우수한 정확도를 통해, CPC는 speaker identity와 음성 콘텐츠를 모두 캡처함을 주장

t-SNE 비교

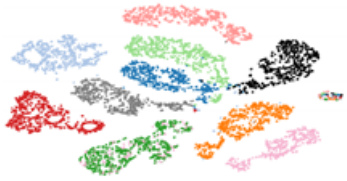


Figure 2: t-SNE visualization of audio (speech) representations for a subset of 10 speakers (out of 251). Every color represents a different speaker.

Speaker identification에 대한 t-SNE 분포

- GRU의 maximum context 크기는 성능에 큰 영향을 미침
- 더 긴 세그먼트는 더 나은 결과를 제공할 수 있음
- 20480개의 wav length (약 1.3초)에 대한 결과임

Summary - Contrastive Predictive Coding

1. 텍스트, 음성, 비디오, 이미지와 같이 정렬된 순서로 표현할 수 있는 모든 형태의 데이터 적용 가능
2. 여러 시간 간격으로 떨어져 있는 데이터 포인트들 $x_{[t,t+k]}$에서 공유되는 정보를 인코딩
   - Representation 학습을 통해 Slow features를 얻고 이를 이용하여 미래의 데이터를 예측함
   - 현재의 target을 Positive로 두고 noise 등이 포함 된 Negative들을 구별할 수 있는 InfoNCE loss를 사용하여 미래의 데이터를 예측하도록 함

3. 단일 테스크에서 여러 k_s를 사용하여 다양한 시간 척도에서 진화하는 feature들을 캡처함
4. x_t에 대한 representation을 계산할 때, encoder 네트워크 위에서 실행되는 Autoregressive Network를 사용하여 과거 context 정보를 encoding 할 수 있음 (미래의 것은 사용 X)

Conclusion & Major take away

1. 최근 이미지 분야의 Self-supervised 에서 널리 사용되는 Contrastive learning에 대하여 분석해보았음
2. 음성에 대해서는 미래 예측 step k에 대해서 사용하고, 이미지에 대해서는 (sequential 하지 않다 보니) 현재의 sample 및 augmented 된 것들이 아닌 다른 class에 대해 negative sample로 간주하여 학습하는 기법임
3. 앞으로 리뷰할 논문 및 최근 트렌드가 해당 내용이므로, 반드시 알고 가야 할 기법이었음
4. 더 자세한 내용은 코드 리뷰할 필요가 있음

Appendix

- Self-supervised learning using contrastive learning (InfoNCE loss), [4]
- [4] 논문은 Contrastive Learning에 대해 잘 리뷰한 논문이고, 해당 논문을 참조하면 더 많은 정보를 얻을 수 있을 것 같음

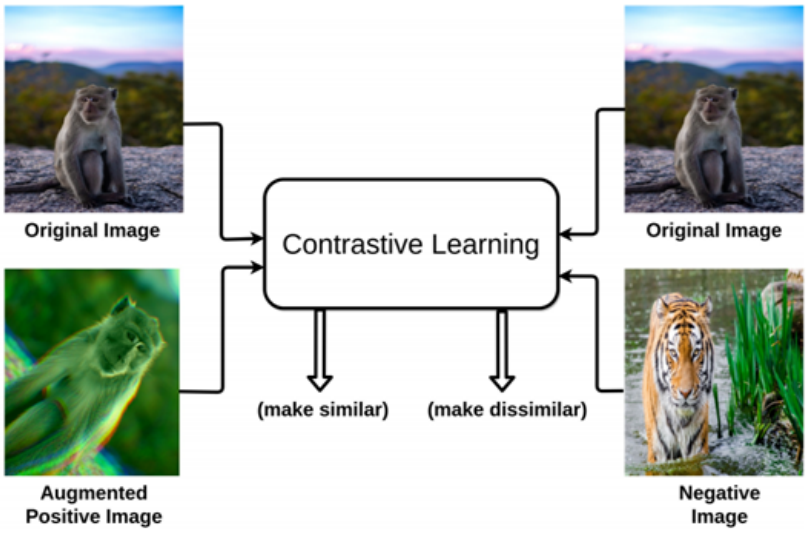


Figure 1. Basic intuition behind contrastive learning paradigm: push original and augmented images closer and push original and negative images away.

1. Contrastive learning paradigm [4]

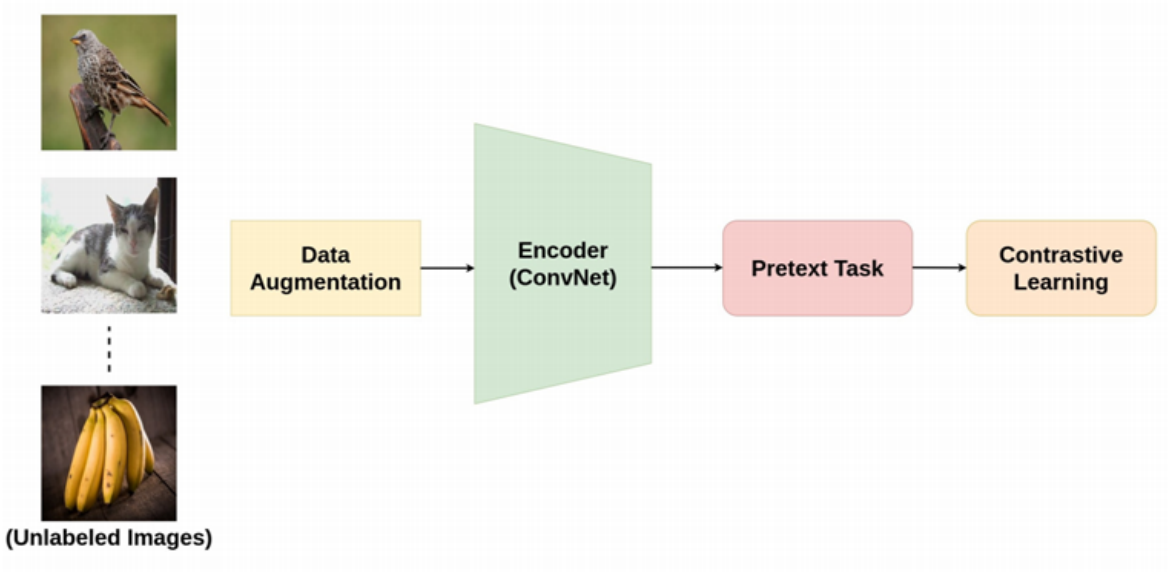


Figure 2. Contrastive learning pipeline for self-supervised training.

2. Network flow of contrastive [4]

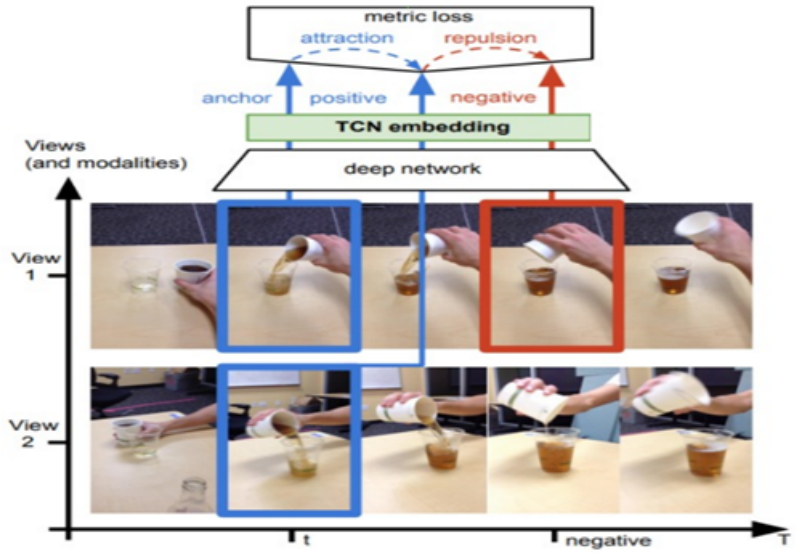
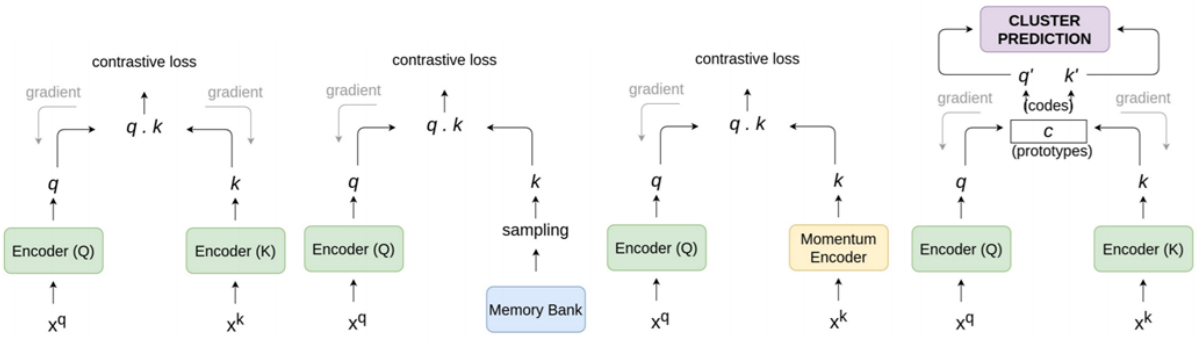


Figure 8. Learning representation from video frame sequence [23].

3. Video processing with contrastive learning [4]



(a) End-to-End (b) Memory Bank (c) Momentum Encoder (d) Clustering

Figure 11. Different architecture pipelines for Contrastive Learning: (**a**) End-to-End training of two encoders where one generates representation for positive samples and the other for negative samples. (**b**) Using a memory bank to store and retrieve encodings of negative samples. (**c**) Using a momentum encoder which acts as a dynamic dictionary lookup for encodings of negative samples during training. (**d**) Implementing a clustering mechanism by using swapped prediction of the obtained representations from both the encoders using end-to-end architecture.

4. Different architecture pipelines for Contrastive Learning [4]

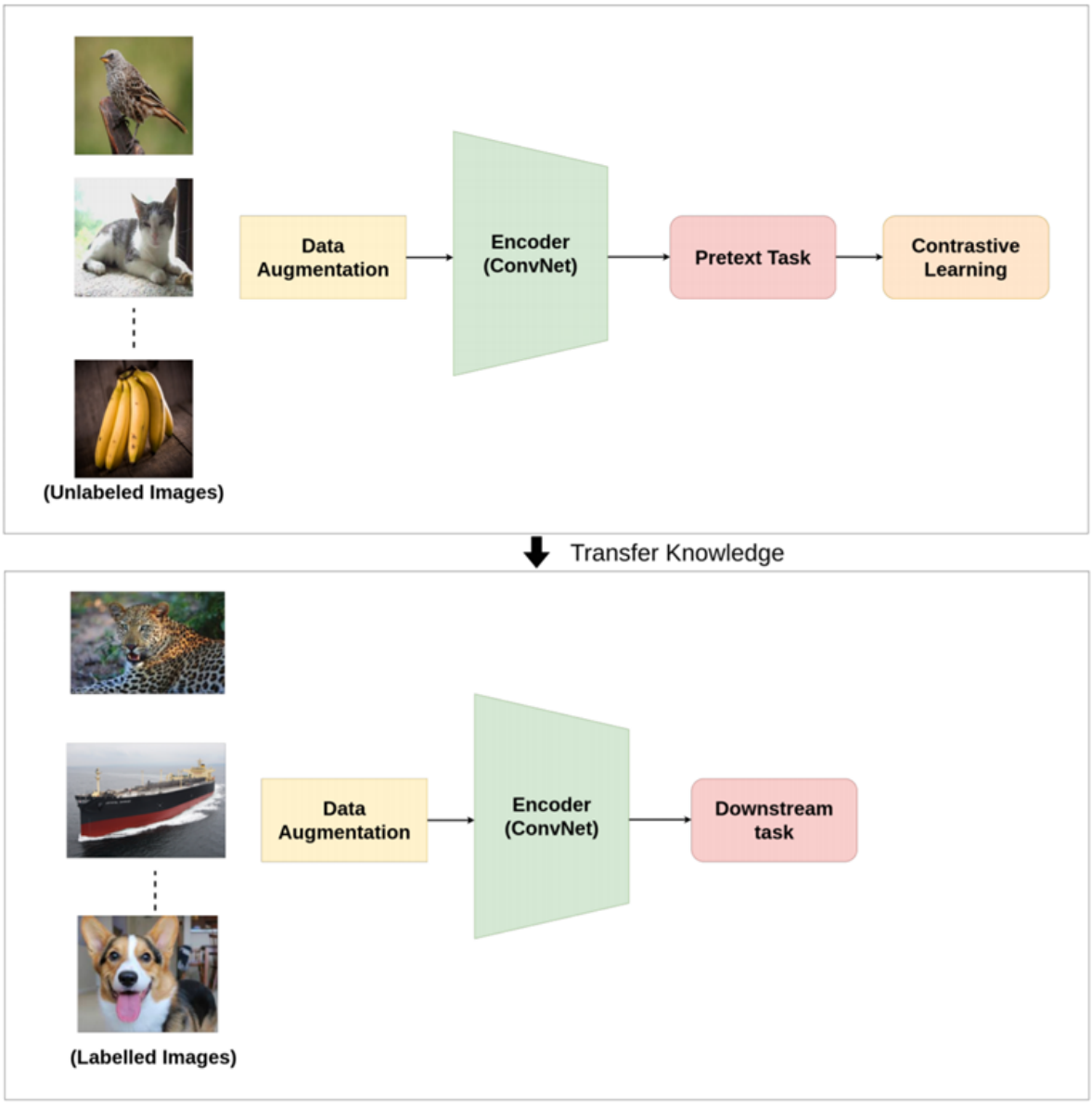


Figure 16. An overview of downstream task for images.

5. Pre-training, fine-tuning of Contrastive learning [4]

5. Training

To train an encoder, a pretext task is used that utilizes contrastive loss for backpropagation. The central idea in contrastive learning is to bring similar instances closer and push away dissimilar instances far from each other. One way to achieve this is to use a similarity metric that measures the closeness between the embeddings of two samples. In a contrastive setup, the most common similarity metric used is cosine similarity that acts as a basis for different contrastive loss functions. The cosine similarity of two variables (vectors) is the cosine of the angle between them and is defined as follows.

$$cos\_sim(A,B) = \frac{A.B}{\|A\|\|B\|} \tag{2}$$

Contrastive learning focuses on comparing the embeddings with a Noise Contrastive Estimation (NCE) [43] function that is defined as

$$L_{NCE} = -log\frac{exp(sim(q,k_+)/\tau)}{exp(sim(q,k_+)/\tau) + exp(sim(q,k\_)/\tau)} \tag{3}$$

where q is the original sample, k_+ represents a positive sample, and $k\_$ represents a negative sample. τ is a hyperparameter used in most of the recent methods and is called temperature coefficient. The sim() function can be any similarity function, but generally a cosine similarity as defined in Equation (2) is used. The initial idea behind NCE was to perform a nonlinear logistic regression that discriminates between observed data and some artificially generated noise.

If the number of negative samples is greater, a variant of NCE called InfoNCE is used as represented in Equation (4). The use of l2 normalization (i.e., cosine similarity) and the temperature coefficient effectively weighs different examples and can help the model learn from hard negatives.

$$L_{infoNCE} = -log\frac{exp(sim(q,k_+)/\tau)}{exp(sim(q,k_+)/\tau) + \sum_{i=0}^{K} exp(sim(q,k_i)/\tau)} \tag{4}$$

where k_i represents a negative sample.

6. Explanation of InfoNCE loss [4]

References

[1] Oord, Aaron van den, Yazhe Li, and Oriol Vinyals. "Representation learning with contrastive predictive coding." arXiv preprint arXiv:1807.03748 (2018).

[2] Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781 (2013).

[3] Gutmann, Michael, and Aapo Hyvärinen. "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models." Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. JMLR Workshop and Conference Proceedings, 2010.

[4] Jaiswal, Ashish, et al. "A survey on contrastive self-supervised learning." Technologies 9.1 (2021): 2.