TTIC 31230, Fundamentals of Deep Learning, Winter 2019

David McAllester

Pretraining

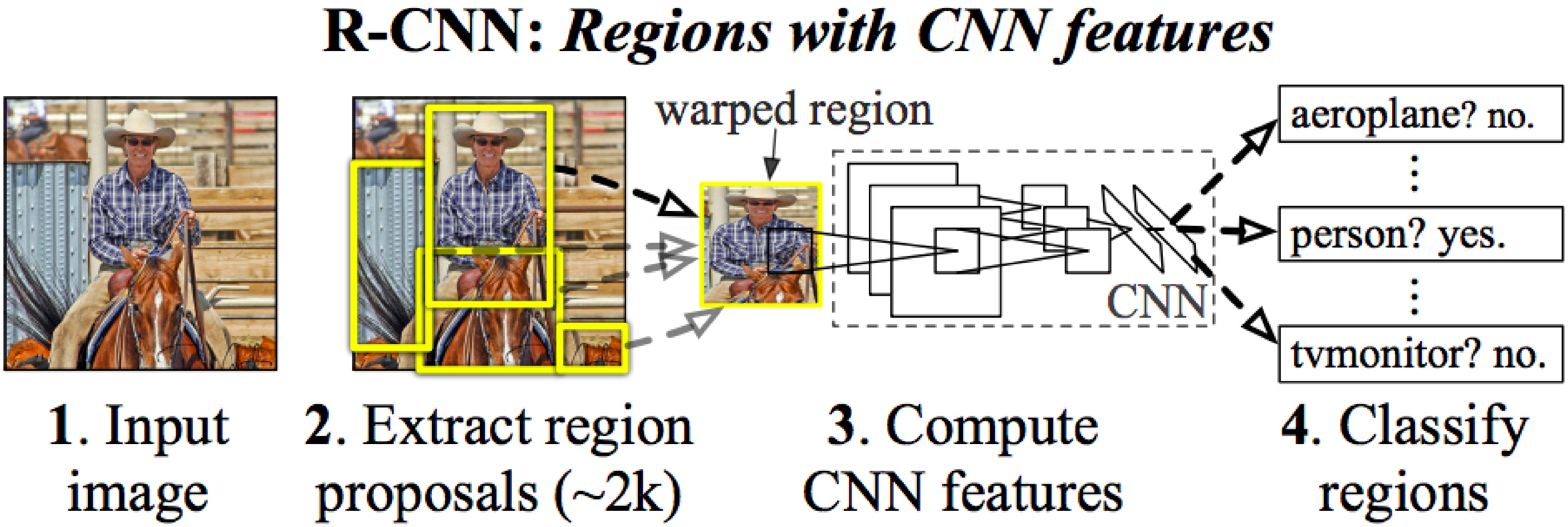
Supervised Pretraining

We train a general purpose model and then fine tune that model on a different task.

This has been standard practice with Imagenet models since 2013.

General Learning Algorithms vs. Learning General Knowledge.

Example: R-CNN, Girshick et al., Nov. 2013



Note that Alexnet appeared on the scene at the ImageNet evaluation (ILSVRC2012) Oct. 2012.

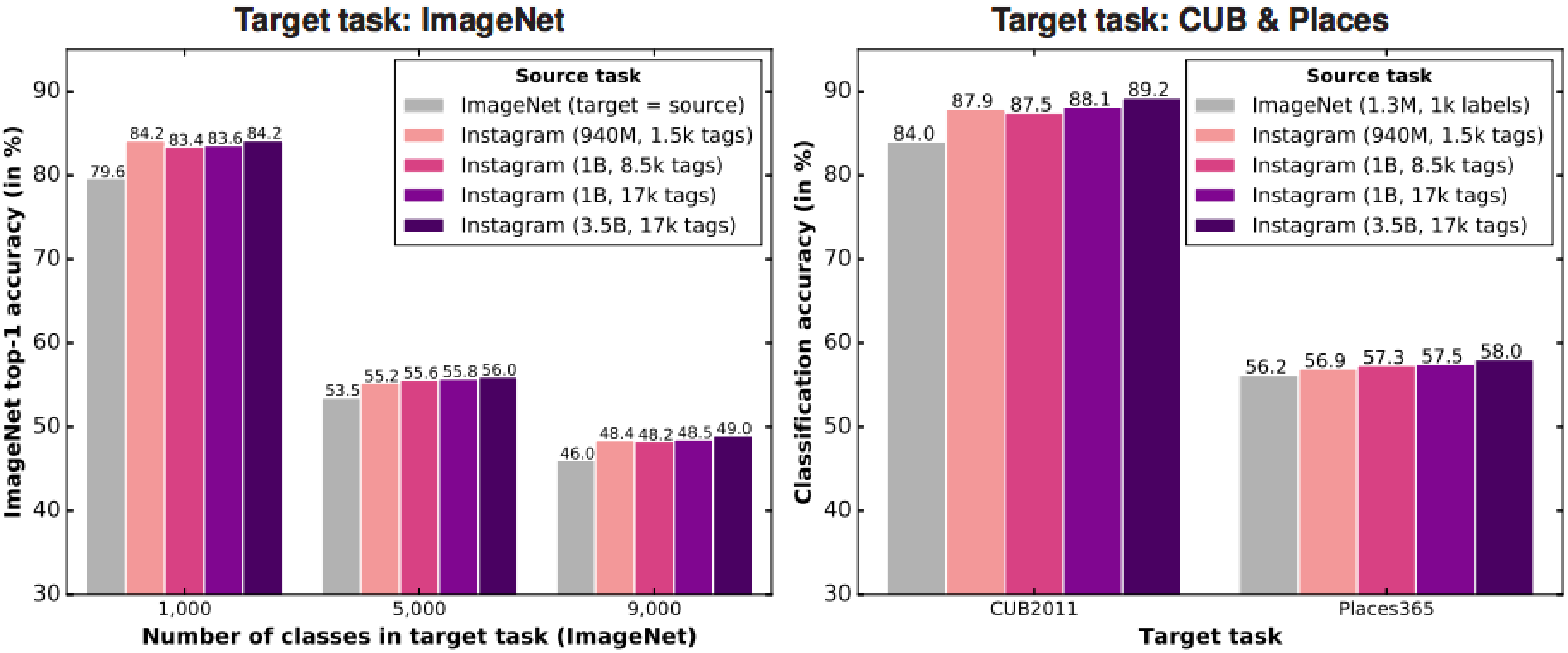
Instagram Pretraining, Mahajan et al., May 2018

In our experiments, we train standard convolutional network architectures to predict hashtags on up to 3.5 billion public Instagram images.

To make training at this scale practical, we adopt a distributed synchronous implementation of stochastic gradient descent with large (8k image) minibatches, following Goyal et al. 2017.

Instagram Pretraining

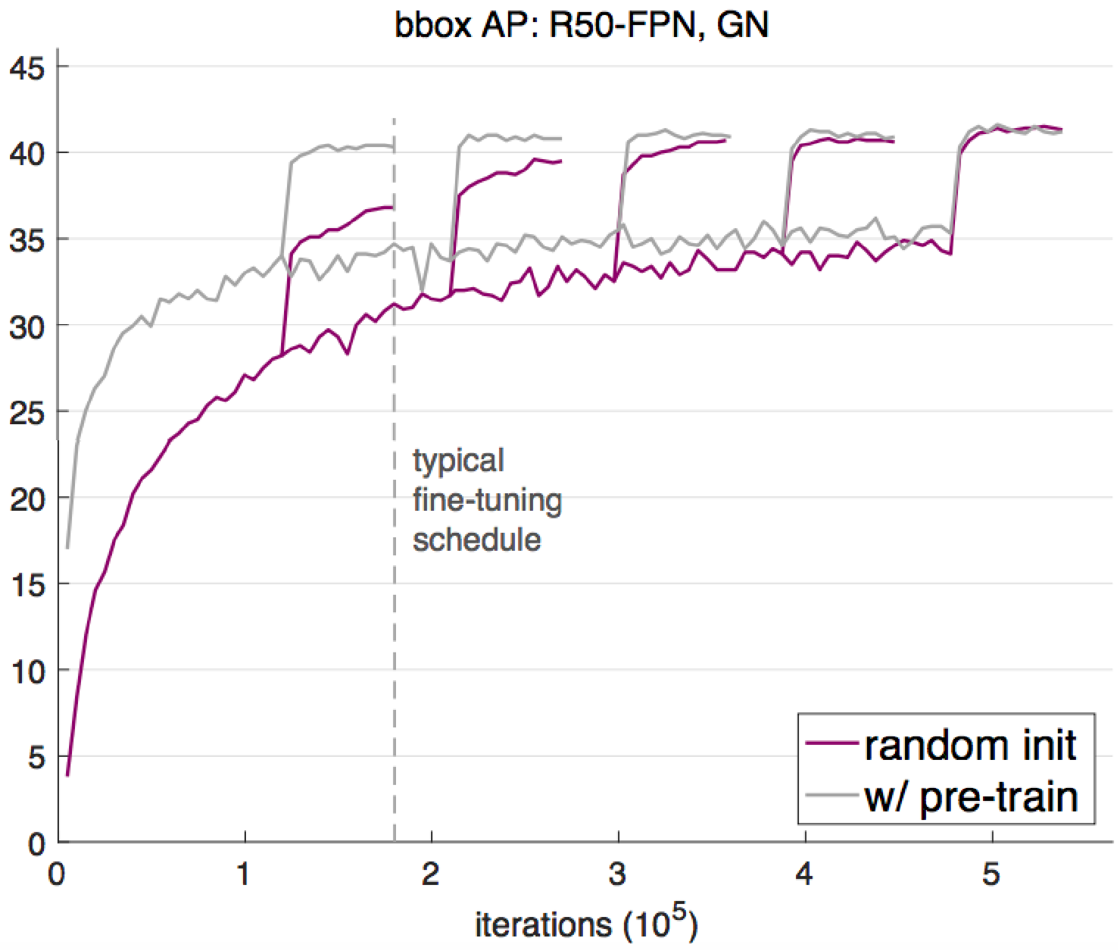
Exploring the Limits of Weakly Supervised Pretraining 7



Rethinking ImageNet Pretraining, He et al., Nov. 2018

We report competitive results on object detection and instance segmentation on the COCO dataset using standard models trained from **random initialization**.

Rethinking ImageNet Pretraining, He et al., Nov. 2018



Self-Supervised Pretraining

We find some naturally occuring source of (x, y) pairs and then treat it as a supervised learning problem.

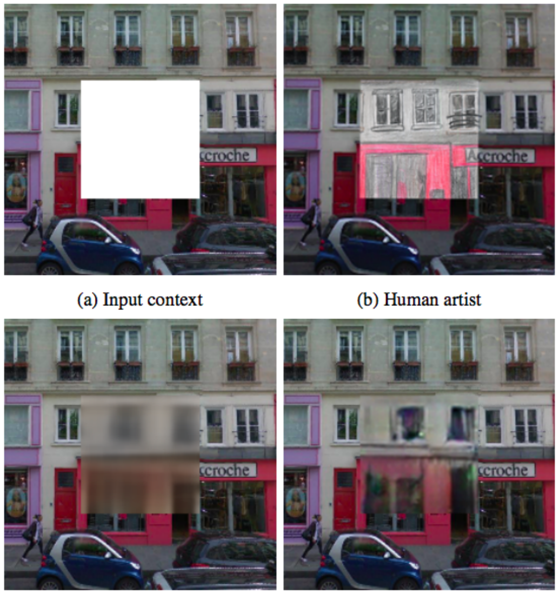
We then initialize learning models the weights derived from pretraining.

We will consider inpainting and colorization.

Feature Learning by Inpainting

Pathak et al., April 2016

$$\Phi^* = \operatorname*{argmin}_{\Phi} \; E_{(x,y)\sim \mathrm{Pop}} \; ||y - \hat{y}_\Phi(x)||^2 + \lambda \mathrm{Discr}(y, \hat{y}_\Phi(x))$$



(a) Input context

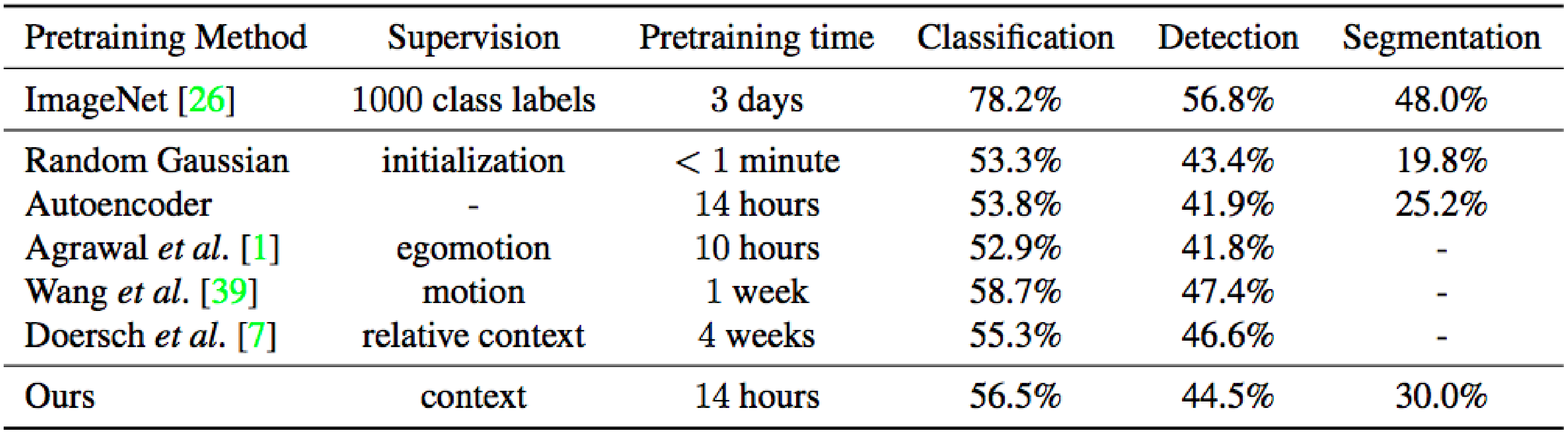
(b) Human artist

(c) Context Encoder ($L2$ loss)

(d) Context Encoder ($L2$ + Adversarial loss)

Feature Learning by Inpainting

Pathak et al., April 2016



Pretraining Method	Supervision	Pretraining time	Classification	Detection	Segmentation
ImageNet [26]	1000 class labels	3 days	78.2%	56.8%	48.0%
Random Gaussian	initialization	< 1 minute	53.3%	43.4%	19.8%
Autoencoder	-	14 hours	53.8%	41.9%	25.2%
Agrawal *et al.* [1]	egomotion	10 hours	52.9%	41.8%	-
Wang *et al.* [39]	motion	1 week	58.7%	47.4%	-
Doersch *et al.* [7]	relative context	4 weeks	55.3%	46.6%	-
Ours	context	14 hours	56.5%	44.5%	30.0%

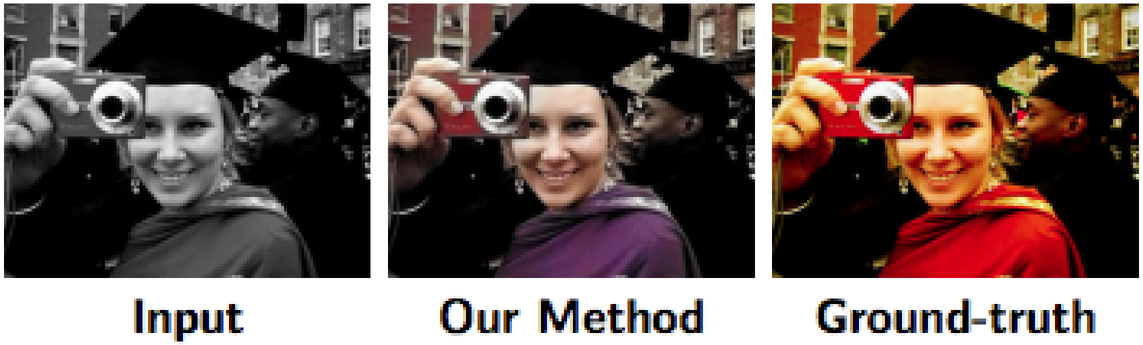
Alexnet, Trained on Imagenet, Evaluated on PASCAL

Image Count: Instagram 3.5G, ImageNet 14M, Pascal 5.7k

Learning Representations for Colorization

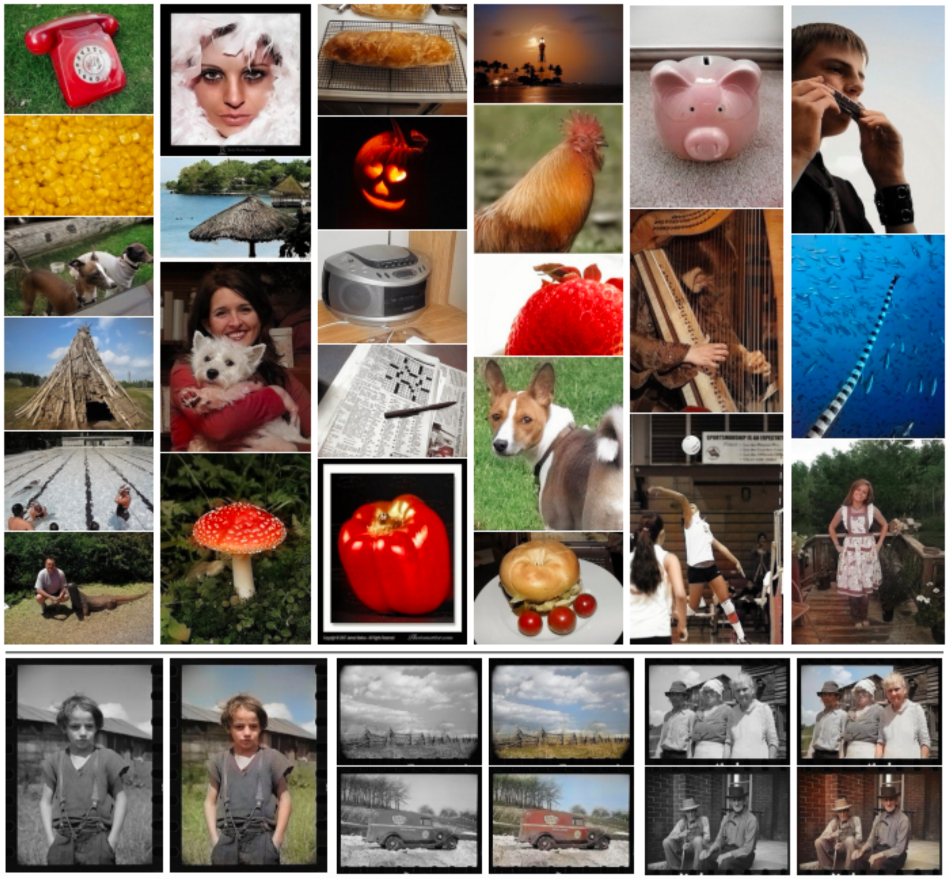
Larsson et al., March 2016

$$\Phi^* = \operatorname*{argmin}_{\Phi} \; E_{x,y\sim\mathrm{Pop}} \, ||y - \hat{y}_\Phi(x)||^2$$



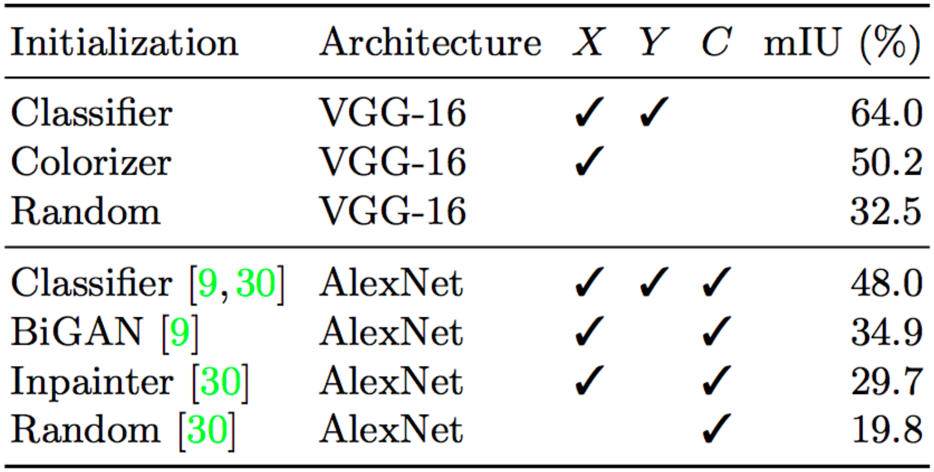
Learning Representations for Colorization

Larsson et al., March 2016



Learning Representations for Colorization

Larsson et al., March 2016



Initialization	Architecture	X	Y	C	mIU (%)
Classifier	VGG-16	✓	✓		64.0
Colorizer	VGG-16	✓			50.2
Random	VGG-16				32.5
Classifier [9, 30]	AlexNet	✓	✓	✓	48.0
BiGAN [9]	AlexNet	✓		✓	34.9
Inpainter [30]	AlexNet	✓		✓	29.7
Random [30]	AlexNet			✓	19.8

VGG-16, Pretrained on ImageNet evaluated on the PASCAL segmentation task. Images Counts: Instagram 3.5G, ImageNet 14M, Pascal 5.7k. X — pretraining used, Y —pretrained with Labels, C — trained on PASCAL with color.

Adding a Discrimination Loss to a Noisy-Channel RDA

$$P_\Phi(y,z) = \mathrm{Pop}(y)P_\Phi(z|y)$$

$$\Phi^* \;=\; \underset{\Phi}{\mathrm{argmin}} \left\{ \begin{array}{l} I(y,z) \\ +\lambda_1\; E_{y,z}\; \mathrm{Dist}(y,\hat{y}_\Phi(z)) \\ +\lambda_2\; E_{y,z}\; \mathrm{Discr}(y,\hat{y}_\Phi(z)) \end{array} \right.$$

This can be viewed as an RDA with a discrimination distortion, or as a GAN with a rate-distortion objective.

Adding a Discrimination Loss
to a Conditional Noisy-Channel RDA

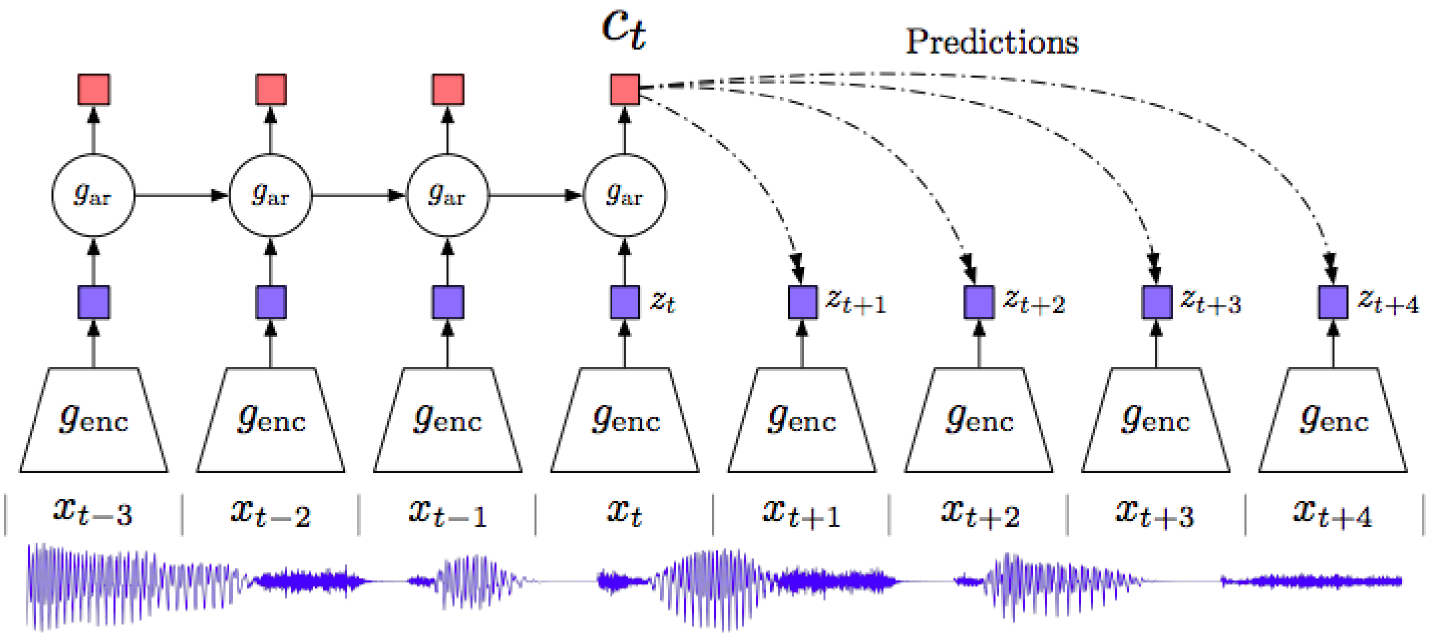
To model a class-conditional image distribution we might use the following where x is a class label and y is an image.

$$P_\Phi(x,y,z) = \mathrm{Pop}(x,y)P_\Phi(z|y,x)$$

$$\Phi^* = \operatorname*{argmin}_\Phi E_{x,y,z} \left\{ \begin{array}{l} KL(P_\Phi(z|y,x), P_\Phi(z|x)) \\ +\lambda_1 \mathrm{Dist}(y,\ \hat{y}_\Phi(z)) \\ +\lambda_2 \mathrm{Discr}(y,\ \hat{y}_\Phi(z) \mid x) \end{array} \right.$$

Contrastive Predictive Coding (CPC)

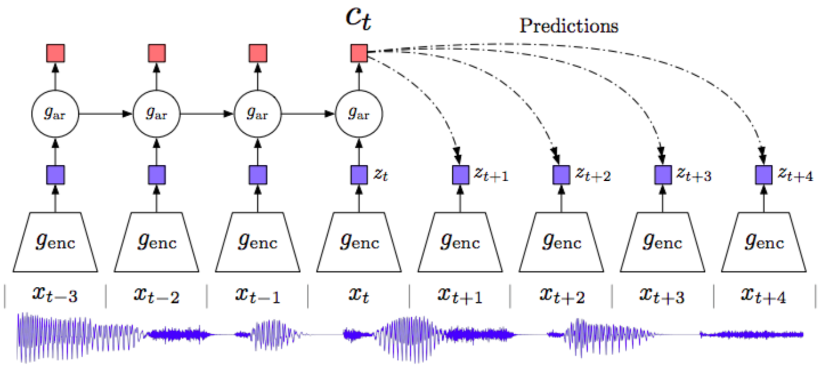
van den Oord et al., July 2018



We $x_1, \ldots x_t$ and make predictions about $x_{t+1}, \ldots, x_{t+k}$

Contrastive Predictive Coding (CPC)

van den Oord et al., July 2018



Rather than predict $x_{t+\Delta t}$ we predict a feature vector $z_\Phi(x_t + \Delta t)$ from the feature vectors $z_\Phi(x_1), \ldots z_\Phi(x_t)$.

We want the feature vector $z_\Phi(x)$ to carry the “signal” and drop the “noise” — more later.

CPC

$$\Phi^* = \operatorname*{argmin}_{\Phi} \;\; E_{x_1,\ldots,x_t,\ldots,x_{t+k}} \sum_{\Delta t} \lambda_{\Delta t} \, \mathcal{L}_{\mathrm{CEN}}(x_{t+\Delta t})$$

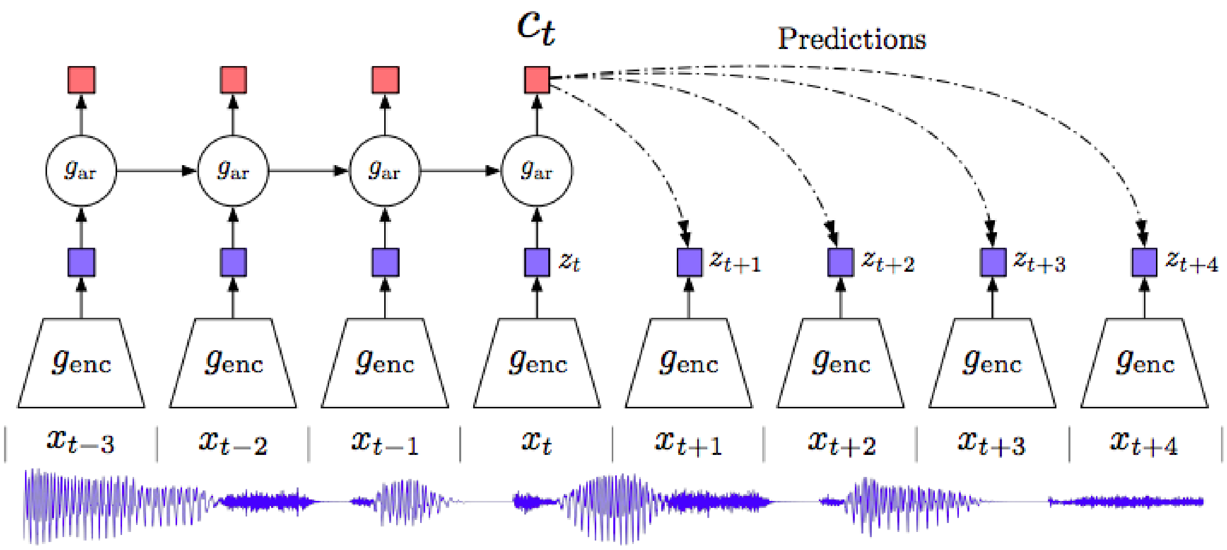
$$\mathcal{L}_{\mathrm{CEN}}(x_{t+\Delta t}) = E_{x_1,\ldots x_N \sim \mathrm{Pop}^N}$$

$$- \ln P_\Phi(i | c_t, \Delta t, z_\Phi(x_{t+\Delta t}) \hookrightarrow z_\Phi(x_1), \ldots, z_\Phi(x_N))$$

$$P_\Phi(i | c_t, \Delta t, z_1, \ldots, z_{N+1}) = \operatorname*{softmax}_{i} \;\; z_i W_{\Delta t} \, c_t$$

The context c_t and learned features z_t must have linearly-extractable information.

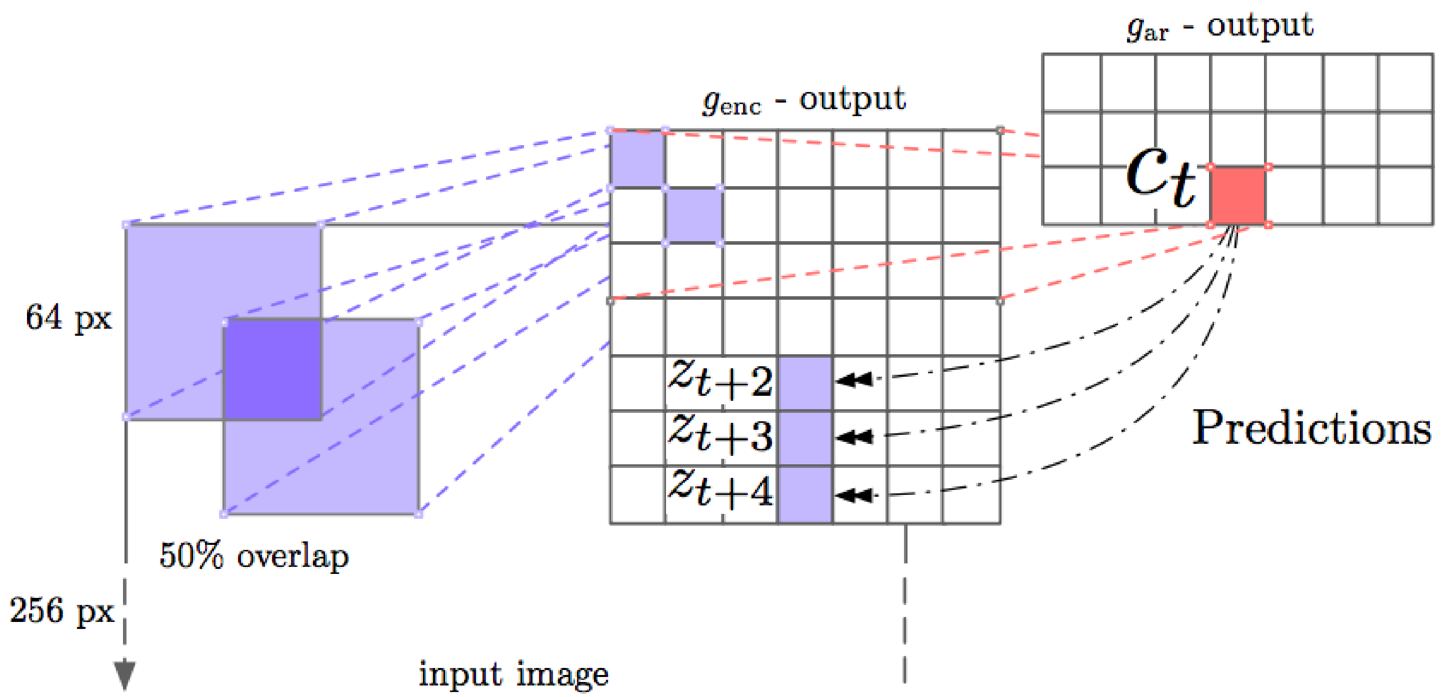
Maximizing a Lower Bound on Mutual Information



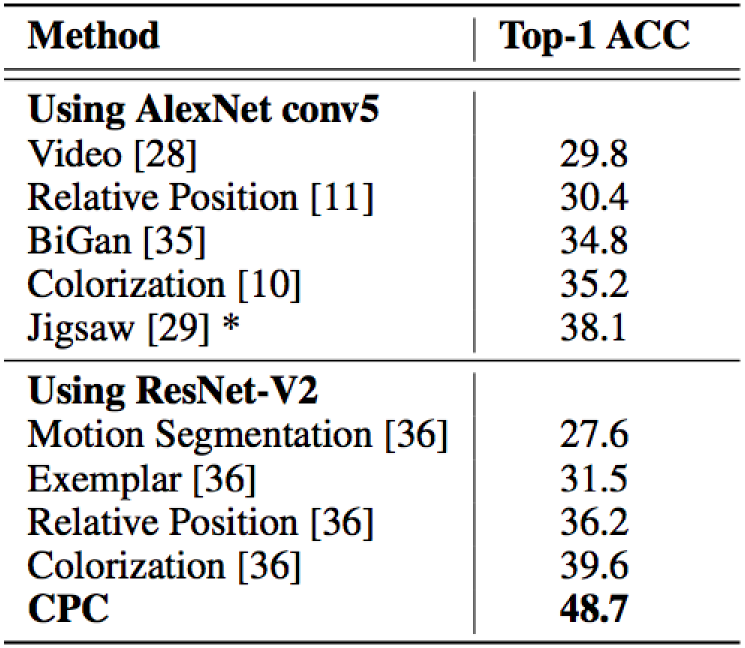
$$I(c_t, x_{t+\Delta t}) \geq \ln(N+1) - \mathcal{L}_{\mathrm{CEN}}(\Delta t)$$

By maximizing mutual information we are intuitively separating signal from noise.

CPC for Vision



CPC Vision Results



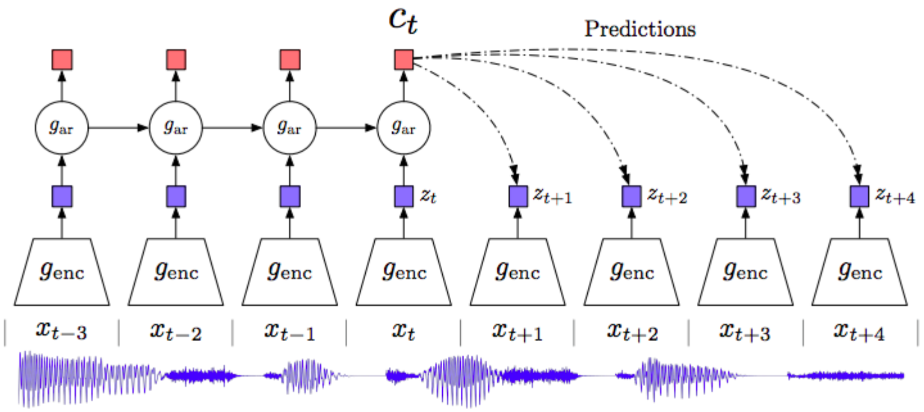
Method	Top-1 ACC
Using AlexNet conv5	
Video [28]	29.8
Relative Position [11]	30.4
BiGan [35]	34.8
Colorization [10]	35.2
Jigsaw [29] *	38.1
Using ResNet-V2	
Motion Segmentation [36]	27.6
Exemplar [36]	31.5
Relative Position [36]	36.2
Colorization [36]	39.6
CPC	**48.7**

Train and test are both on Imagenet but labels are only used retraining the final linear layer.

Results are also give for speech recognition tasks and NLP tasks.

Information Theoretic Co-Training

McAllester, Feb. 2018



$$\Phi^* = \operatorname*{argmax}_{\Phi} \left(\sum_{\Delta t} \lambda_{\Delta t}\, \hat{I}(c, z_\Phi(x_{t+\Delta t}))\right) - \lambda \hat{H}(z_\Phi(x))$$

$$\hat{I}(c, z_\Phi(x_{t+\Delta t})) = \hat{H}(z_\Phi(x)) - \hat{H}(z_\Phi(x_{t+\Delta t})|c)$$

Difference of Entropy Models

$$\hat{I}(c, z_\Phi(x_{t+\Delta t})) = \hat{H}(z_\Phi(x)) - \hat{H}(z_\Phi(x_{t+\Delta t})|c)$$

We maximize the mutual information estimate where the entropy estimates are done with cross-entropy models — upper bounds on true entropy.

Note that maximizing $\hat{H}(z_\Phi(x))$ is adversarial — Φ wants to increase the cross entropy while the cross entropy model is trying to minimize the cross entropy.

Pretraining for NLP

- Word Embeddings

- The Transformer

- ELMO, BERT and GPT-2

Advances in Pre-Training Distributed Word Representations

Mikolov et al., 2017

We want a mapping from a word w to a vector $e(w)$ — a word embedding.

We can also consider phrase embedings $e(P)$ for a word sequence P.

fastText from Facebook is currently popular.

It provides both contextual bag of words (cbow) and byte pair encoding (BPE) word vectors.

cbow word vectors

We construct a population distribution on pairs (c, w) here c is a bag of word context and w is a word.

$$\Phi^* = \operatorname*{argmin}_{\Phi} E_{c,w} \;\; - \ln P(w|c)$$

Φ consists of a matrix $e[w, i]$ where $e[w, I]$ is the word embedding of w, and a matrix $e'[w, i]$ giving the embedding of the word w when it appears in a context.

A score $s(w|c)$ is defined by

$$s(w|c) = \frac{1}{|c|} \sum_{w' \in c} \; e(w)^\top e'(w')$$

Negative Sampling in cbow

Rather than define $P_\Phi(w|c)$ by a softmax over w, one uses restricted negative sampling.

We construct a training set of triples (w, c, N_C)

$$\Phi^* = \operatorname*{argmin}_\Phi \; E_{w,c,N_c} \; \ln\left(1 + e^{-s(w,c)}\right) + \sum_{n \in N_C} \ln\;\left(1 + e^{s(n,c)}\right)$$

Byte Pair Encoding (BPE)

BPE constructs a set of character n-grams by starting with the unigrams and then greedily merging most common bigrams of n-grams.

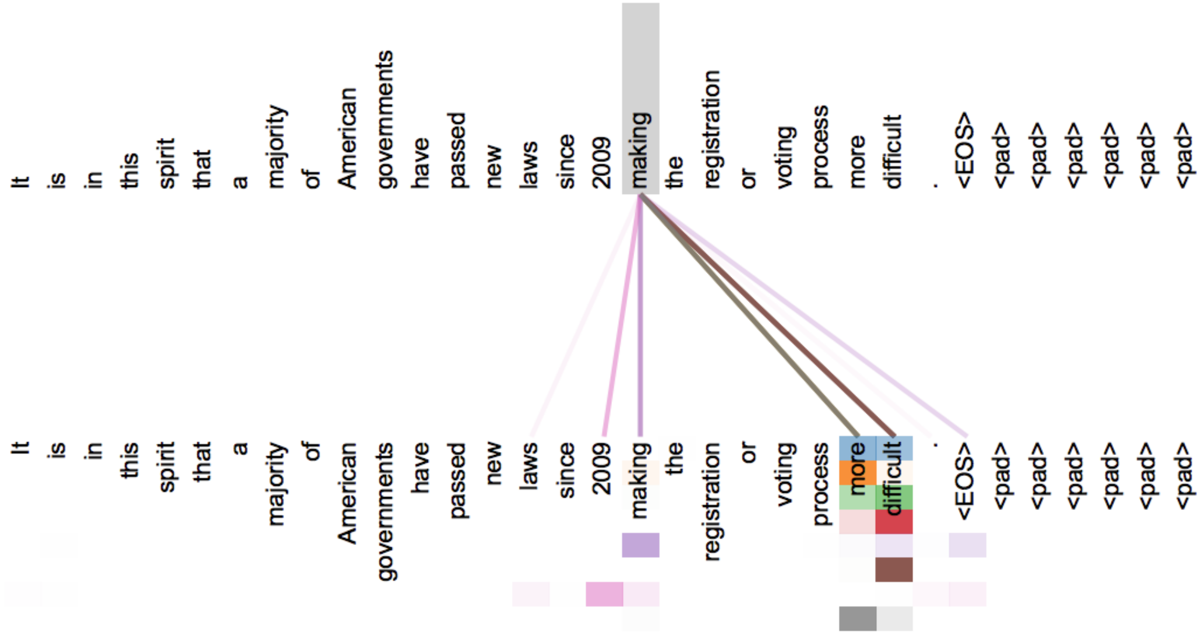
Given a set of character n-grams each word is treated as a bag of character n-grams.

$$e[w] = \frac{1}{N} \sum_{n \in w} e(n)$$

Attention is All You Need (the Transformer)

Vaswani et al., June 2017

The Transformer is based on multi-headed self-attention.



Review of Attention

Procedure Lookup(key$[J]$, $M[T, J]$)

$$s[t] = \mathrm{key}[J]^\top M[t, J]$$

$$\alpha[t] = \operatorname*{softmax}_t s[t] \text{ ; the attention}$$

$$V[J] = \sum_t \alpha[t] M[t, J]$$

Return $V[J]$

Multi-Layer, Multi-Headed Attention

Let ℓ ranger over layers — each layer will be computed from the previous layer.

Let h range over "heads" — each head is computing an attention for a different purpose.

We will construct layers of sequences of vectors $M[L, T, J]$.

Multi-Headed Self Attention

To compute $M[\ell+1,T,J]$ from $M[\ell,T,J]$ we first compute:

$$\mathrm{Query}[\ell,h,t,k] = \sum_j W^Q[\ell,h,k,j]M[\ell,t,j]$$

$$\mathrm{Key}[\ell,h,t,k] = \sum_j W^K[\ell,h,k,j]M[\ell,t,j]$$

$$\mathrm{Value}[\ell,h,t,i] = \sum_j W^V[\ell,h,i,j]M[\ell,t,j]$$

Multi-Headed Self Attention

We then compute $M[\ell+1,T,J]$ as follows.

$$s[\ell,h,t,t'] = \frac{1}{\sqrt{K}}\sum_k \mathrm{Query}[\ell,h,t,k]\mathrm{Key}[\ell,h,t',k]$$

$$\alpha[\ell,h,t,t'] = \operatorname*{softmax}_{t'} \; s[\ell,h,t,t'] \quad \text{the attention}$$

$$V[\ell,h,t,i] = \sum_{t'} \alpha[\ell,h,t,t']\mathrm{Value}[\ell,h,t',i]$$

$$M[\ell{+}1,t,J] = V[\ell,1,t,I];V[\ell,2,t,I];\cdots;V[\ell,H,t,I] \qquad J = HI$$

Encoding Positional Information

At the input layer we augment the word embeddings with position information. For example:

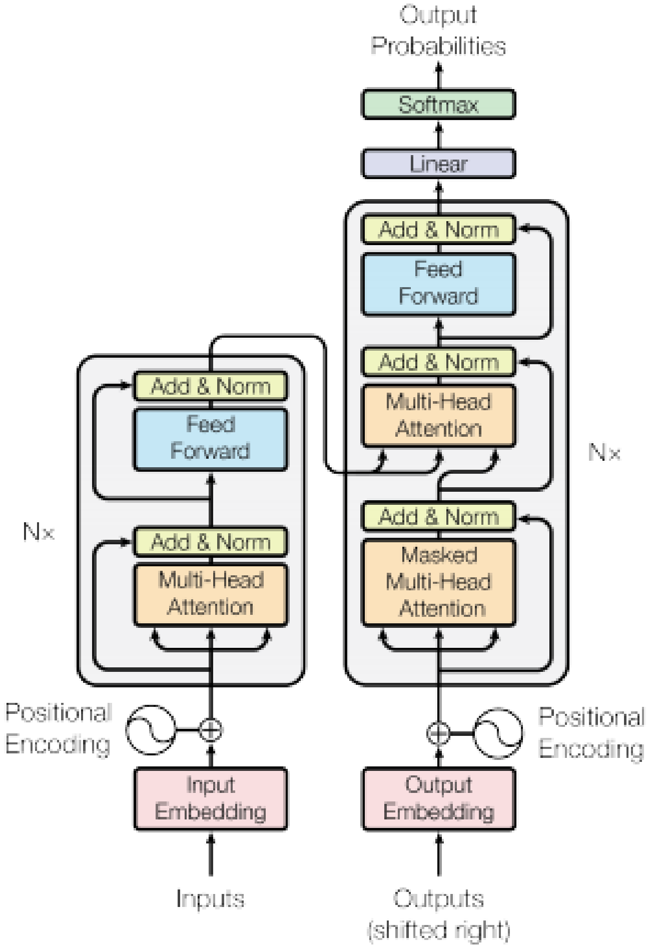
$$M[0,t,J] = e[w[t],I]; e^{i\omega t}\ ; e^{i2\omega t}\ ; e^{i4\omega t} \cdots\ ; e^{i2^k\omega t}$$

This is analogous to adding a k-bit binary representation of t.

Each complex number can be interpreted as a pointer on a dial.

Each dial rotates twice as fast as the dial before.

The Transformer as a Translation Model



Deep Contextualized Word Representations (ELMo)

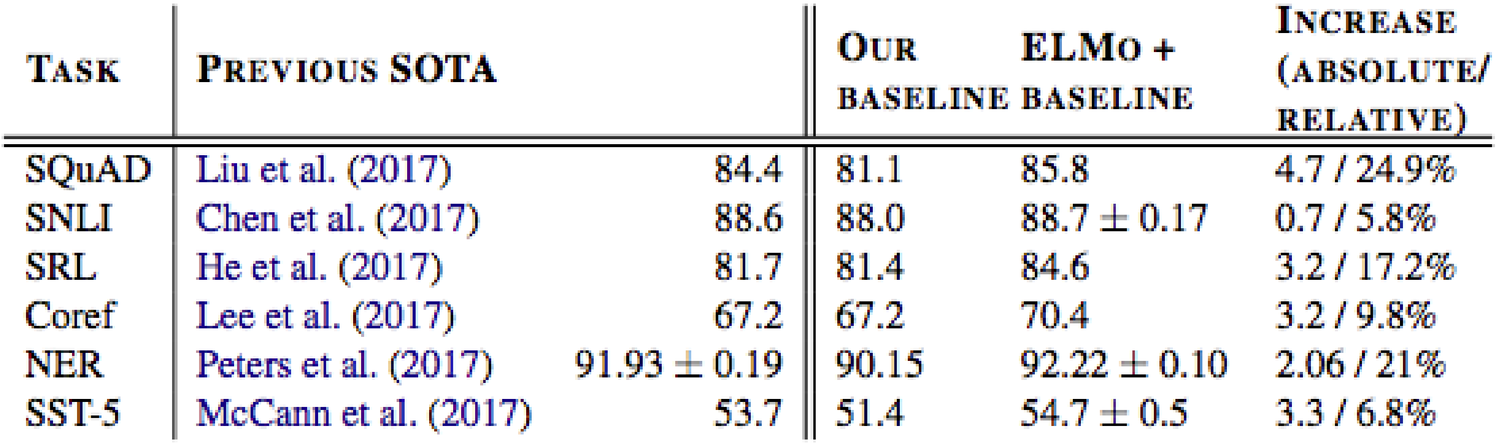
Peters et al., Feb. 2018 (AI2)

ELMo (EMbeddings from Language models) is a two layer bidirectional LSTM character language model trained to predict the next word in both the forward and backward direction.

It is designed to be used to generate “contextual word embeddings” as inputs to any system that uses word embeddings.

Deep Contextualized Word Representations (ELMo)

Peters et al., Feb. 2018 (AI2)



Task	Previous SOTA		Our baseline	ELMo + baseline	Increase (absolute/ relative)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%

BERT: Pre-training of Deep Bidirectional Transformers

Devlin et al., Oct. 2018 (Google AI Language)

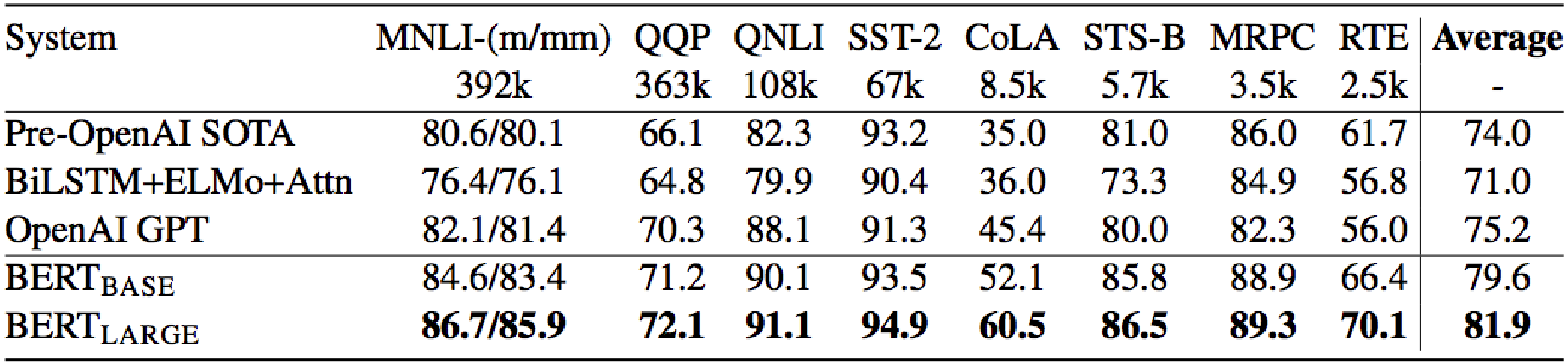
BERT (Bidirectional Encoder Representations from Transformers) is Transformer language models trained to fill in sparse blanks that have been placed in text.

It also generates sentence embeddings and has a second "skip thought" training criterion where it predicts the next senentence embedding (thought vector) from the previous one.

It is designed to be used by adding a single task specific output layer and then fine tuning the entire model to the task.

BERT: Pre-training of Deep Bidirectional Transformers

Devlin et al., Oct. 2018 (Google AI Language)



System	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	**Average**
	392k	363k	108k	67k	8.5k	5.7k	3.5k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.9	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	88.1	91.3	45.4	80.0	82.3	56.0	75.2
BERT$_{\text{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
BERT$_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

BERT results on the GLUE benchmark problems (General Language Understanding Evaluation) excluding the Winograd problems. The first problem (MNLI – Multi genre Natual Langauge Inference) has two metrics. The number below each problem is the training set size.

GLUE tasks

Sentiment Analysis Given a sentence does it express positive or negative sentiment (toward a product in a product review). SST-2.

Natural Langauge Inference (NLI). Given a pair of sentences classify the relationship into "entailed", "contradictory" or "netural". MNLI, QNLI, RTE.

Semantic Equivalence. Given two sentences do they express the same meaning. QQP, STS-B, MRPC.

Grammaticality Judgements. Given a sentence is it grammatical. CoLA.

Winograd Problems. Determine the referent of a pronoun in cases requiring semantics. WNLI

Language Models [are] Multitask Learners, (GPT-2)
Radford et al., Feb. 2019 (Open AI)

This is a Transformer language model trained on a new text corpus, webText, scraped from the internet and containing 40G words of text.

Historically sampling text from a language model has yielded semantic gibberish. This time things are different.

Continuing from a Prompt

Continue from:

In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

The Predicted Continuation

Continuation excerpted from a single response, the response selected from 10 tries.

The scientist named the population, after their distinctive horn, Ovids Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

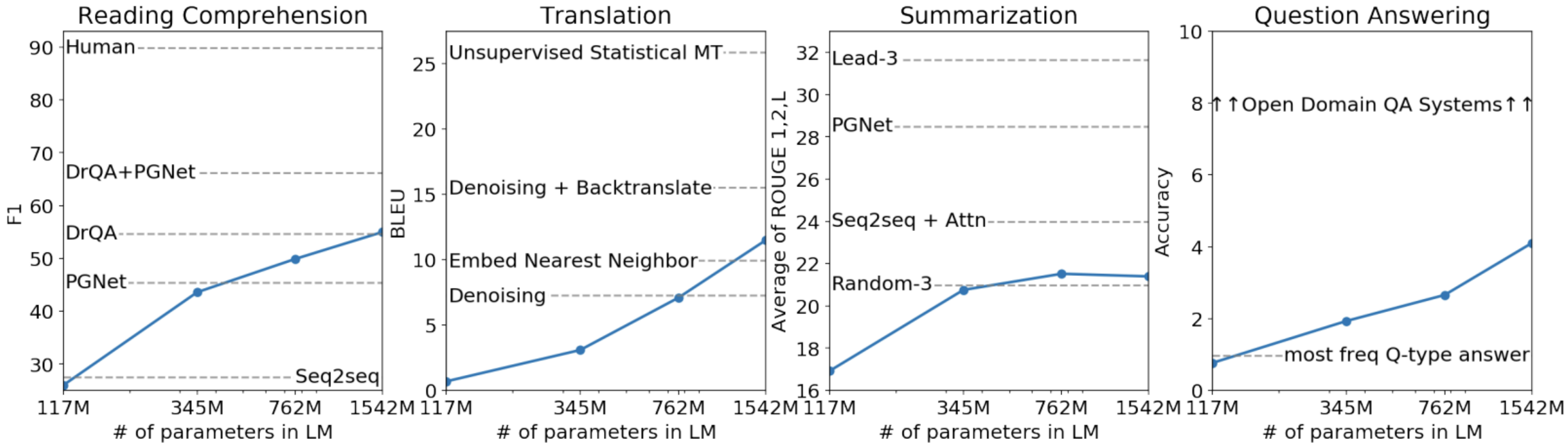
Dr. Jorge Prez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when ... Prez and his friends were astonished to see the unicorn herd. ... While examining these bizarre creatures the scientists discovered that the creatures also spoke some fairly regular English. Prez stated, We can see, for example, that they have a common language, something like a dialect or dialectic.

Dr. Prez believes that the unicorns may have originated in Argentina ... some believe that perhaps the creatures were created when a human and a unicorn met each other in a time before human civilization. ... However, Prez also pointed out that it is likely that the only way of knowing for sure if unicorns are indeed the descendants of a lost alien race is through DNA. ...

A Laguage Model Translating French to English

Continue from: <French1> = <English1>, ... ,<Frenchn> = <Englishn>, <French> =

Note that the translation pairs in the prompt do not influence the model parameters.



END