DSC291: Machine Learning with Few Labels

Large Language Models
Self-Supervised Learning

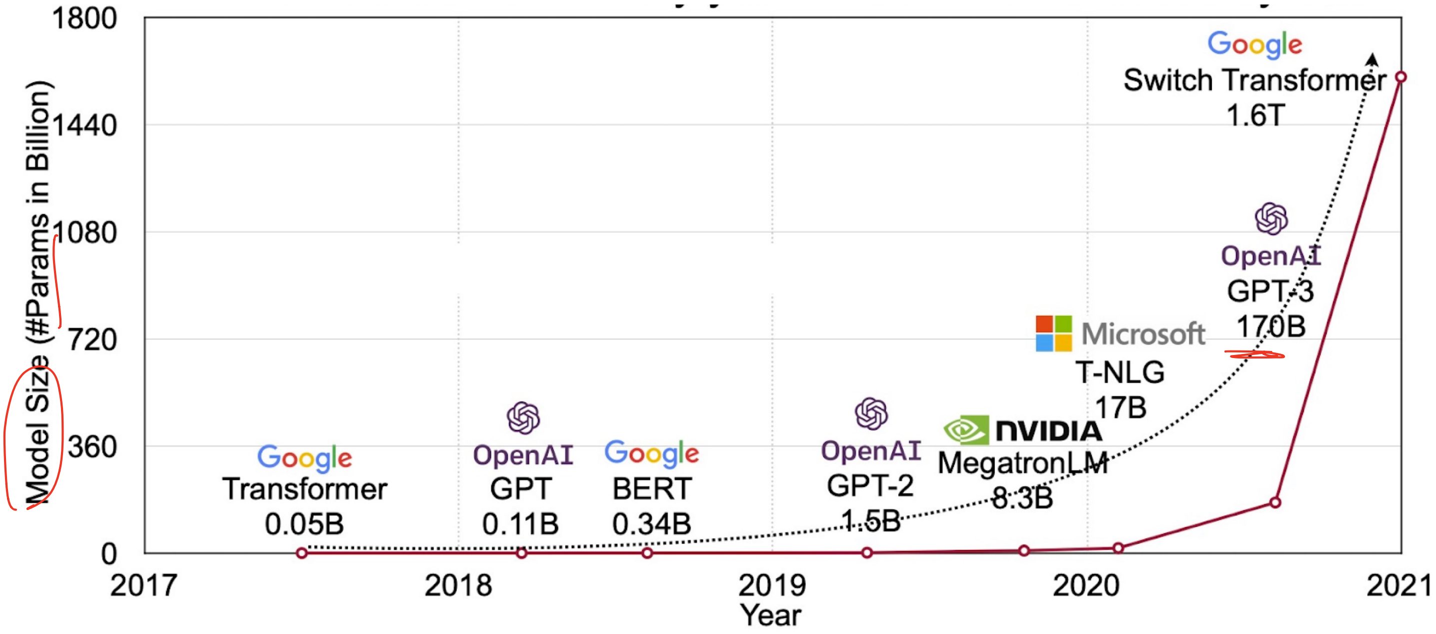
Zhiting Hu

Lecture 5, April 10, 2024

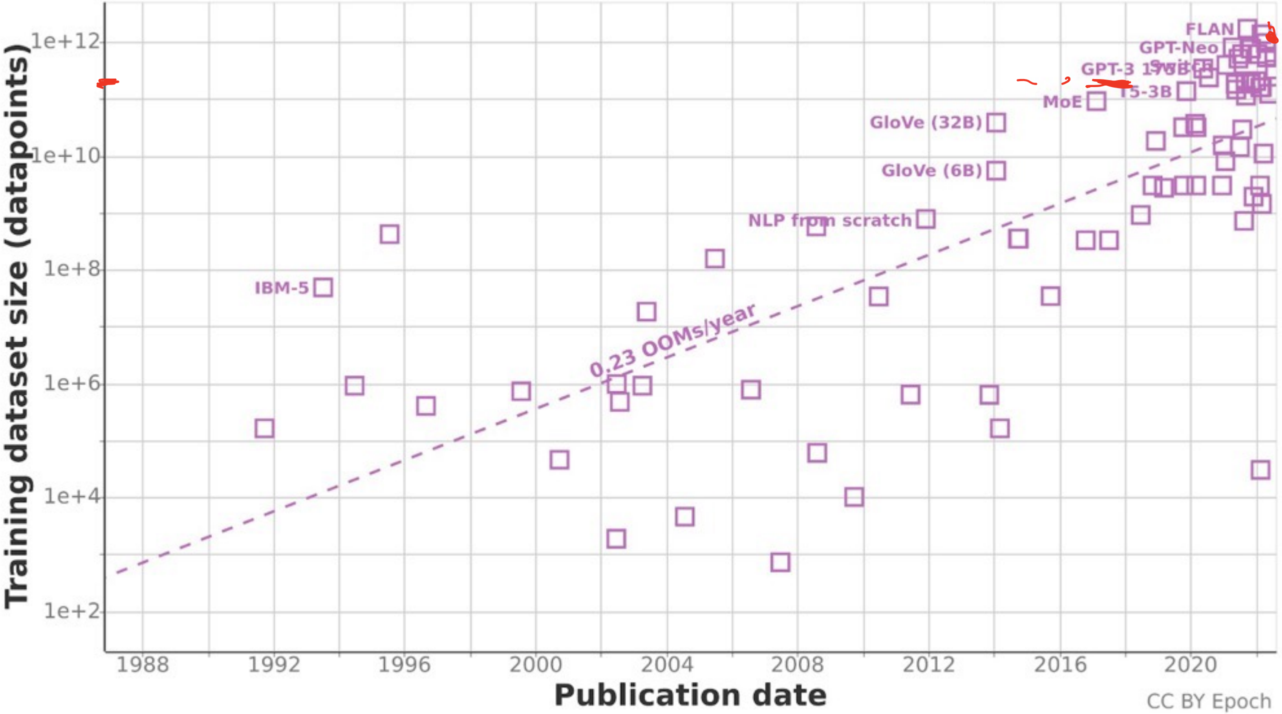
UC San Diego
HALICIOĞLU DATA SCIENCE INSTITUTE

Large Language Models:
More model parameters

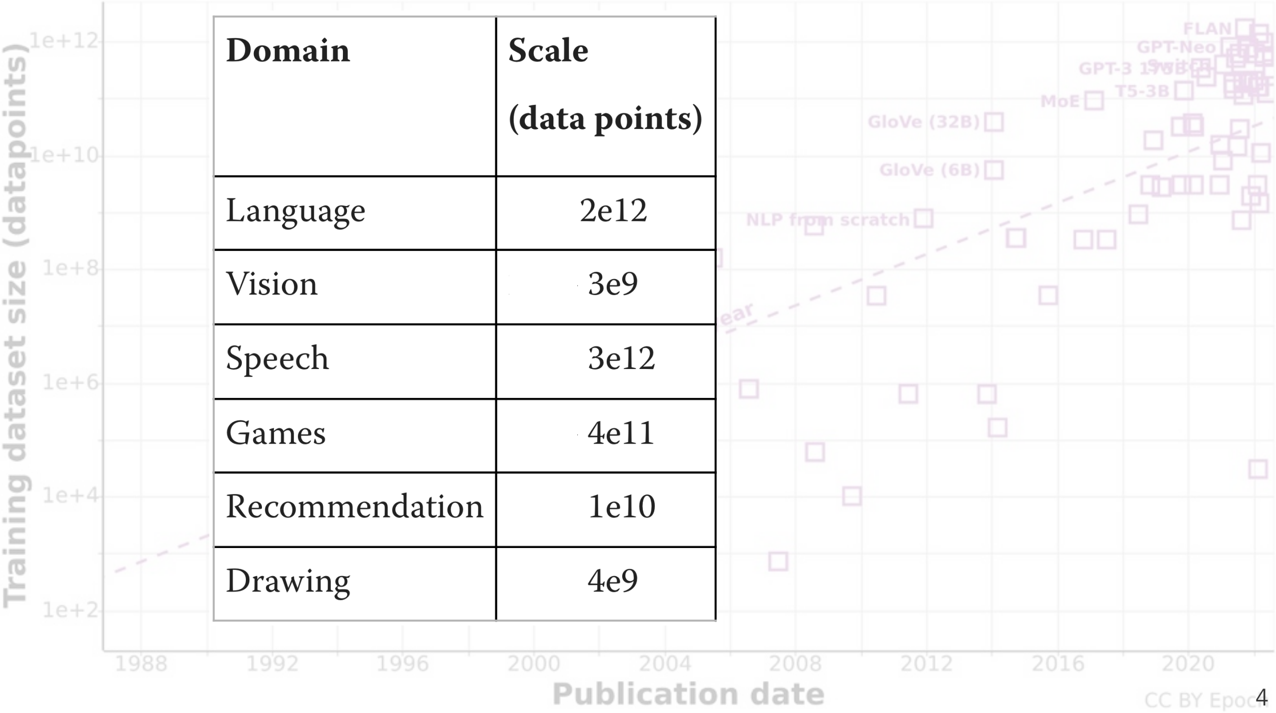
NLP's Moore's Law: Every year model size increases by 10x



Large Language Models:
More model parameters, more data



Large Language Models:
More model parameters, more data



Domain	Scale (data points)
Language	2e12
Vision	3e9
Speech	3e12
Games	4e11
Recommendation	1e10
Drawing	4e9

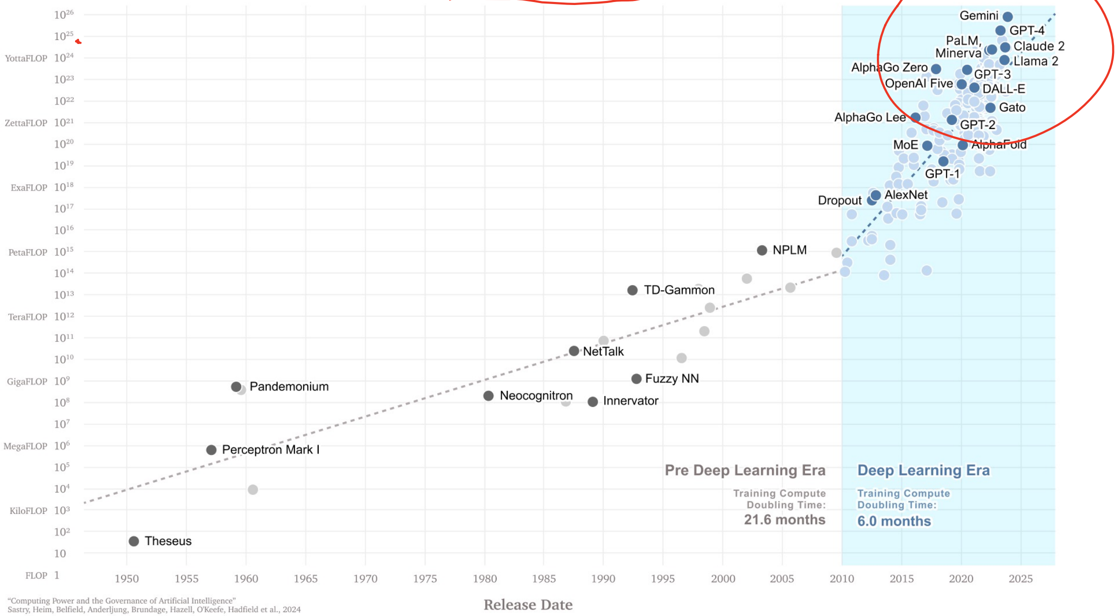
Large Language Models:
More model parameters, more data, more computing

FLOP

10^{25}

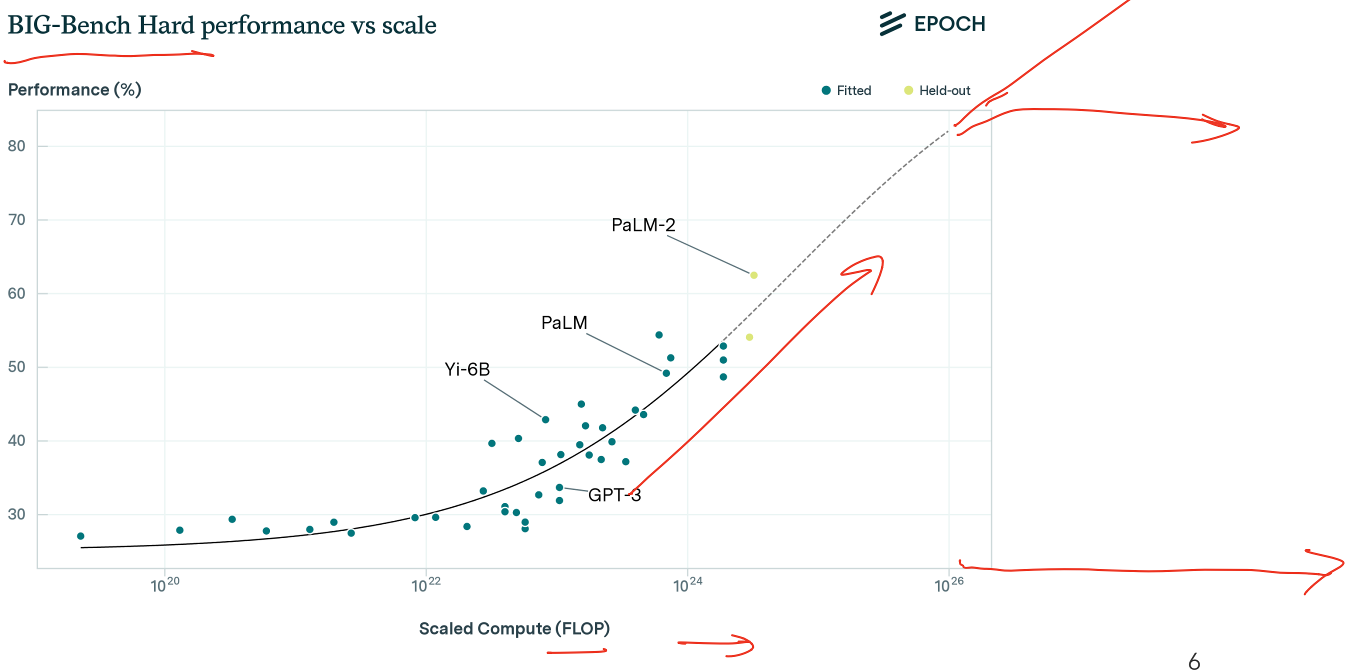
Compute Used for AI Training Runs

Total compute used to train notable AI models, measured in total FLOP (floating-point operations) | Logarithmic

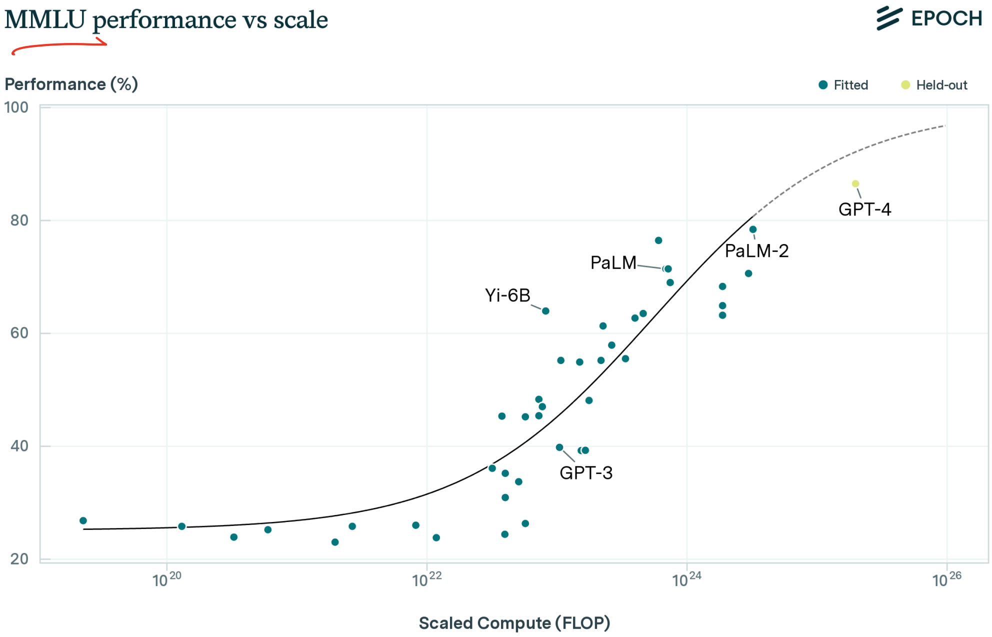


"Computing Power and the Governance of Artificial Intelligence"
Sastry, Heim, Belfield, Anderljung, Brundage, Hazell, O'Keefe, Hadfield et al., 2024

Large Language Models:
More model parameters, more data, more computing



Large Language Models:
More model parameters, more data, more computing



Language models: Summary so far

- So far, we've talked about the model architectures and inference of LMs
  - Model architecture: Transformers
  - Inference: next word prediction (sampling tokens at each step)
- Next: training of LMs

ML solution:

$$\min_{\theta} \mathcal{L}(\theta, \mathcal{E})$$

Self-Supervised Learning

Terminology

- Supervised Learning
- Semi-supervised Learning
- Weakly-supervised Learning
- Self-supervised Learning
- Unsupervised Learning

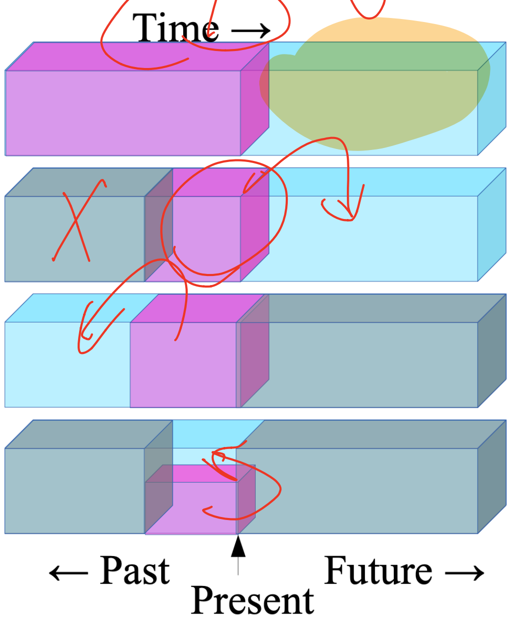
super
semi
un -

$L(\; , \; \varepsilon)$

- All need some forms of supervision, or experience

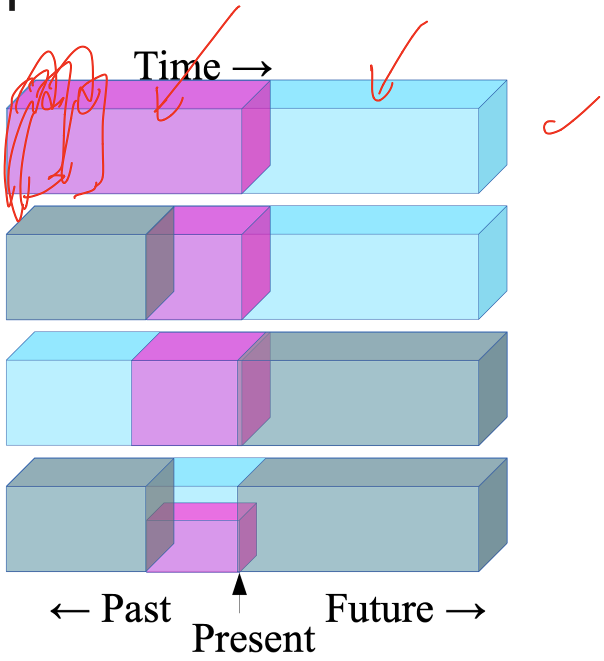
Self-Supervised Learning: Examples

- Predict any part of the input from any other part.
- Predict the future from the past.
- Predict the future from the recent past.
- Predict the past from the present.
- Predict the top from the bottom.



Self-Supervised Learning: Examples

- **Predict any part of the input from any other part.**
- **Predict the future from the past.**
- **Predict the future from the recent past.**
- **Predict the past from the present.**
- **Predict the top from the bottom.**
- **Predict the occluded from the visible**
- **Pretend there is a part of the input you don't know and predict that.**



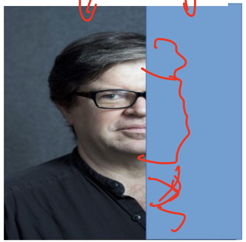
[Courtesy: Lecun "Self-supervised Learning"]

Self-Supervised Learning: Motivation (I)

- Our brains do this all the time

- Filling in the visual field at the retinal blind spot
- Filling in occluded images, missing segments in speech
- Predicting the state of the world from partial (textual) descriptions
- Predicting the consequences of our actions
- Predicting the sequence of actions leading to a result

- Predicting any part of the past, present or future percepts from whatever information is available.



[Courtesy: Lecun "Self-supervised Learning"]

Self-Supervised Learning: Motivation (I)

- Successfully learning to predict everything from everything else would result in the accumulation of lots of background knowledge about how the world works

- The model is forced to learn what we really care about, e.g. a semantic representation, in order to solve the prediction problem

[Courtesy: Lecun “Self-supervised Learning”]

[Courtesy: Zisserman “Self-supervised Learning”]

Self-Supervised Learning: Motivation (II)

- The machine predicts any part of its input from any observed part
  - A lot of supervision signals in each data instance
- Untapped/availability of vast numbers of unlabeled text/images/videos..
  - Facebook: one billion images uploaded per day
  - 300 hours of video are uploaded to YouTube every minute

inference

data, computing

model

[Courtesy: Zisserman "Self-supervised Learning"]

SSL in Language Models

- Calculates the probability of a sentence:
  - Sentence:

$$\boldsymbol{y} = (y_1, y_2, \dots, y_T)$$

$$p_\theta(\boldsymbol{y}) = \prod_{t=1}^{T} \boxed{p_\theta(y_t \mid \boldsymbol{y}_{1:t-1})}$$

Example:

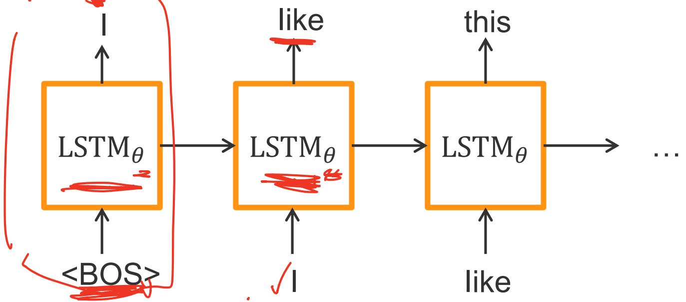
(I, like, this, …)

$$\cdots p_\theta(like \mid I)\, p_\theta(this \mid I, like) \cdots$$

$P(I \mid \text{<BOS>})$

n-gram

Model: LSTM RNN



SSL in Language Models

- Calculates the probability of a sentence:
  - Sentence:

$$\boldsymbol{y} = (y_1, y_2, \dots, y_T)$$

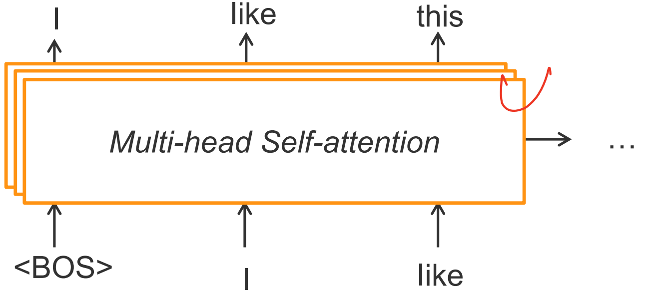
$$p_\theta(\boldsymbol{y}) = \prod_{t=1}^{T} p_\theta(y_t \mid \boldsymbol{y}_{1:t-1})$$

Example:

(I, like, this, …)

$$\cdots p_\theta\,(like \mid I)\, p_\theta(this \mid I, like) \cdots$$

Model: Transformer



SSL in Language Models: Training

- Given data example $\boldsymbol{y}^*$
- Minimizes negative log-likelihood of the data

MLE

$$\min_\theta \mathcal{L}_{\text{MLE}} = \underline{-\log p_\theta(\boldsymbol{y}^*)} = \underline{-\prod_{t=1}^{T} p_\theta(y_t^* \mid \boldsymbol{y}_{1:t-1}^*)}$$

SSL in Language Models: GPT3

- A Transformer-based LM with 125M to 175B parameters
- Trained on massive text data

Dataset	# Tokens (Billions)
Total	499
Common Crawl (filtered by quality)	410
WebText2	19
Books1	12
Books2	55
Wikipedia	3

Brown et al., 2020 "Language Models Are Few-Shot Learners"

[Table from https://lambdalabs.com/blog/demystifying-gpt-3/]

Other examples of self-supervised learning (SSL)

- Learning contextual text representations
- Learning image / video representations

Word Embedding

- Conventional word embedding:
  - Word2vec, Glove
  - A pre-trained **matrix**, each row is an embedding vector of a word

10 yrs ago

test-of-time

60k

60k x 256

256

1000

	0	1	2	3	4	5	6	7	8	9	...
fox	-0.348680	-0.077720	0.177750	-0.094953	-0.452890	0.237790	0.209440	0.037886	0.035064	0.899010	...
ham	-0.773320	-0.282540	0.580760	0.841480	0.258540	0.585210	-0.021890	-0.463680	0.139070	0.658720	...
brown	-0.374120	-0.076264	0.109260	0.186620	0.029943	0.182700	-0.631980	0.133060	-0.128980	0.603430	...
beautiful	0.171200	0.534390	-0.348540	-0.097234	0.101800	-0.170860	0.295650	-0.041816	-0.516550	2.117200	...
jumps	-0.334840	0.215990	-0.350440	-0.260020	0.411070	0.154010	-0.386110	0.206380	0.386700	1.460500	...
eggs	-0.417810	-0.035192	-0.126150	-0.215930	-0.669740	0.513250	-0.797090	-0.068611	0.634660	1.256300	...
beans	-0.423290	-0.264500	0.200870	0.082187	0.066944	1.027600	-0.989140	-0.259950	0.145960	0.766450	...
sky	0.312550	-0.303080	0.019587	-0.354940	0.100180	-0.141530	-0.514270	0.886110	-0.530540	1.556600	...
bacon	-0.430730	-0.016025	0.484620	0.101390	-0.299200	0.761820	-0.353130	-0.325290	0.156730	0.873210	...
breakfast	0.073378	0.227670	0.208420	-0.456790	-0.078219	0.601960	-0.024494	-0.467980	0.054627	2.283700	...
toast	0.130740	-0.193730	0.253270	0.090102	-0.272580	-0.030571	0.096945	-0.115060	0.484000	0.848380	...
today	-0.156570	0.594890	-0.031445	-0.077586	0.278630	-0.509210	-0.066350	-0.081890	-0.047986	2.803600	...
blue	0.129450	0.036518	0.032298	-0.060034	0.399840	-0.103020	-0.507880	0.076630	-0.422920	0.815730	...
green	-0.072368	0.233200	0.137260	-0.156630	0.248440	0.349870	-0.241700	-0.091426	-0.530150	1.341300	...
kings	0.259230	-0.854690	0.360010	-0.642000	0.568530	-0.321420	0.173250	0.133030	-0.089720	1.528600	...
dog	-0.057120	0.052685	0.003026	-0.048517	0.007043	0.041856	-0.024704	-0.039783	0.009614	0.308416	...
sausages	-0.174290	-0.064869	-0.046976	0.287420	-0.128150	0.647630	0.056315	-0.240440	-0.025094	0.502220	...
lazy	-0.353320	-0.299710	-0.176230	-0.321940	-0.385640	0.586110	0.411160	-0.418680	0.073093	1.486500	...
love	0.139490	0.534530	-0.252470	-0.125650	0.048748	0.152440	0.199060	-0.065970	0.128830	2.055900	...
quick	-0.445630	0.191510	-0.249210	0.465900	0.161950	0.212780	-0.046480	0.021170	0.417660	1.686900	...

20 rows × 300 columns

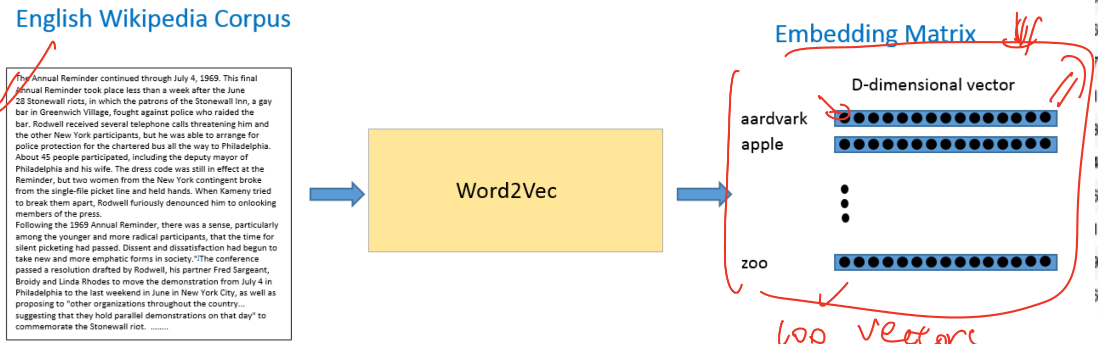
[Courtesy: Vaswani, et al., 2017]

Word Embedding

- Conventional word embedding:
  - Word2vec, Glove
  - A pre-trained **matrix**, each row is an embedding vector of a word

	0	1	2	3	4	5	6	7	8	9	...
fox	-0.348680	-0.077720	0.177750	-0.094953	-0.452890	0.237790	0.209440	0.037886	0.035064	0.899010	...
ham	-0.773320	-0.282540	0.580760	0.841480	0.258540	0.585210	-0.021890	-0.463680	0.139070	0.658720	...
brown	-0.374120	-0.076264	0.109260	0.186620	0.029943	0.182700	-0.631980	0.133060	-0.128980	0.603430	...
beautiful	0.171200	0.534390	-0.348540	-0.097234	0.101800	-0.170860	0.295650	-0.041816	-0.516550	2.117200	...
jumps	-0.334840	0.215990	-0.350440	-0.260020	0.411070	0.154010	-0.386110	0.206380	0.386700	1.460500	...
eggs	-0.417810	-0.035192	-0.126150	-0.215930	-0.669740	0.513250	-0.797090	-0.068611	0.634660	1.256300	...
beans	-0.423290	-0.264500	0.200870	0.082187	0.066944	1.027600	-0.989140	-0.259950	0.145960	0.766450	...
sky	0.312550	-0.303080	0.019587	-0.354940	0.100180	-0.141530	-0.514270	0.886110	-0.530540	1.556600	...
bacon	-0.430730	-0.016025	0.484620	0.101390	-0.299200	0.761820	-0.353130	-0.325290	0.156730	0.873210	...
breakfast	0.073378	0.227670	0.208420	-0.456790	-0.078219	0.601960	-0.024494	-0.467980	0.054627	2.283700	...
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-0.115060	0.484000	0.848380	...
								-0.081890	-0.047986	2.803600	...
								0.076630	-0.422920	0.815730	...
								-0.091426	-0.530150	1.341300	...
								0.133030	-0.089720	1.528600	...
								-0.039783	0.009614	0.308416	...
								-0.240440	-0.025094	0.502220	...
								-0.418680	0.073093	1.486500	...
								-0.065970	0.128830	2.055900	...
								0.021170	0.417660	1.686900	...

100



100 vectors

[Image source: Va

Word Embedding

- **Problem**: word embeddings are applied in a context free manner

`open a bank account`　　　`on the river bank`

`[0.3, 0.2, -0.8, …]`

Courtesy: Devlin 2019

Word Embedding

- **Problem**: word embeddings are applied in a context free manner

`open a bank account`　　　`on the river bank`

`[0.3, 0.2, -0.8, …]`

- **Solution**: Train contextual representations on text corpus

`[0.9, -0.2, 1.6, …]`　　　`[-1.9, -0.4, 0.1, …]`

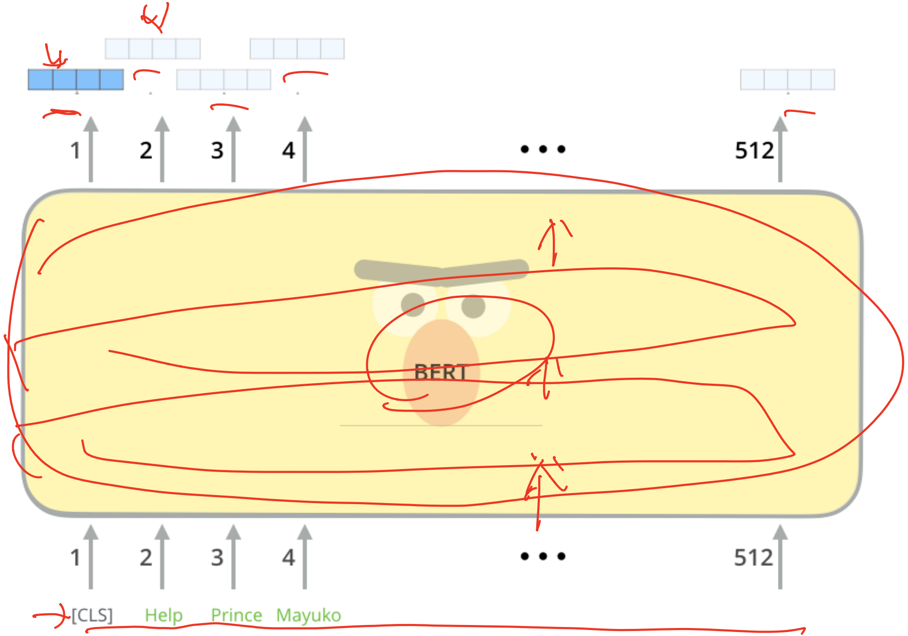
`open a bank account`　　　`on the river bank`

Courtesy: Devlin 2019

Elmo , GPT-1

BERT

- BERT: A bidirectional model to extract contextual word embedding



BERT: Pre-training Procedure

- Dataset:
  - Wikipedia (2.5B words) + a collection of free ebooks (800M words)

BERT: Pre-training Procedure

- Dataset:
  - Wikipedia (2.5B words) + a collection of free ebooks (800M words)

SSL

- Training: **masked language model** (masked LM)
  - Masks some percent of words from the input and has to reconstruct those words from context

BERT: Pre-training Procedure

- Masked LM

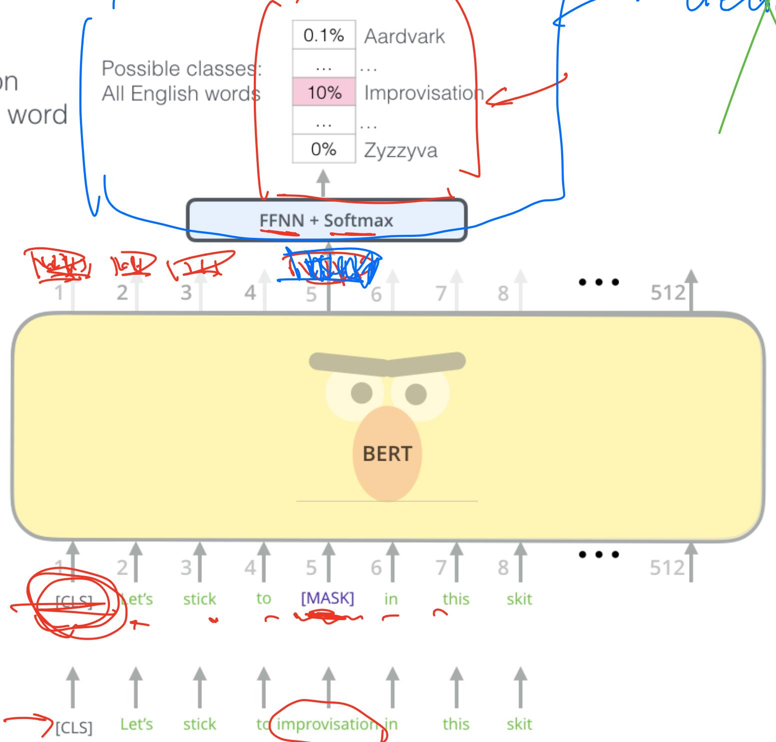
Use the output of the masked word's position to predict the masked word

Randomly mask 15% of tokens

Input

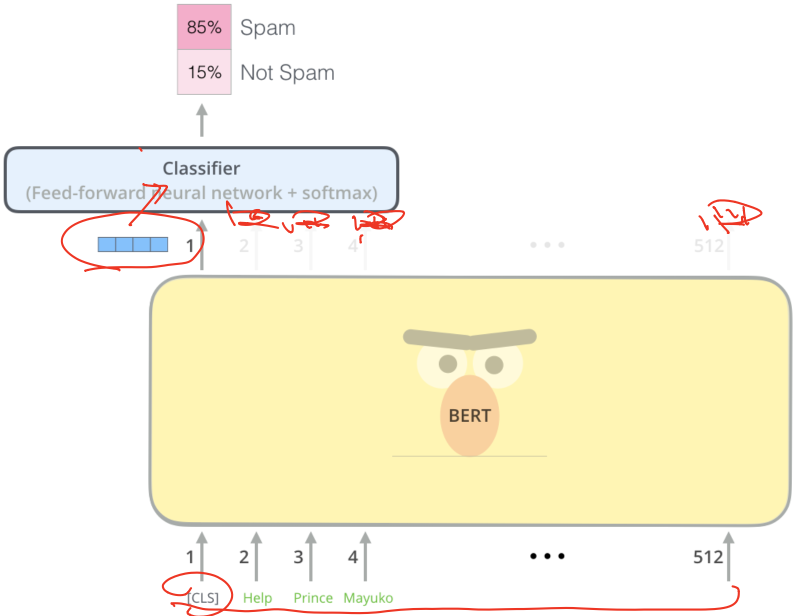
encoder

decoder

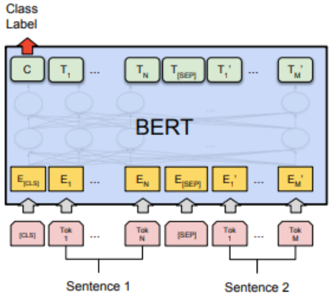


BERT: Downstream Fine-tuning

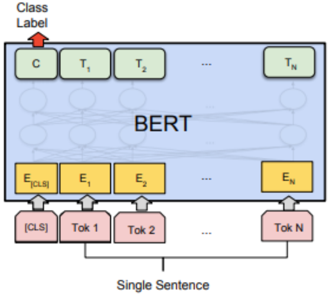
- Use BERT for sentence classification



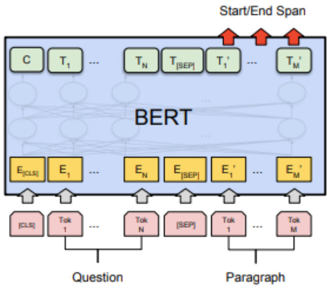
BERT: Downstream Fine-tuning



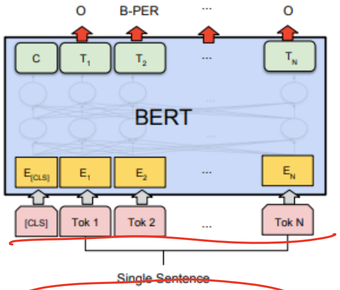
(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



(b) Single Sentence Classification Tasks:
SST-2, CoLA



(c) Question Answering Tasks:
SQuAD v1.1



(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

NER

BERT Results

- Huge improvements over SOTA on 12 NLP task

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
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

Table 1: GLUE Test results, scored by the GLUE evaluation server. The number below each task denotes the number of training examples. The "Average" column is slightly different than the official GLUE score, since we exclude the problematic WNLI set. OpenAI GPT = (L=12, H=768, A=12); $\text{BERT}_{\text{BASE}}$ = (L=12, H=768, A=12); $\text{BERT}_{\text{LARGE}}$ = (L=24, H=1024, A=16). BERT and OpenAI GPT are single-model, single task. All results obtained from https://gluebenchmark.com/leaderboard and https://blog.openai.com/language-unsupervised/.

SSL from Images, EX (I): masked autoencoder (MAE)

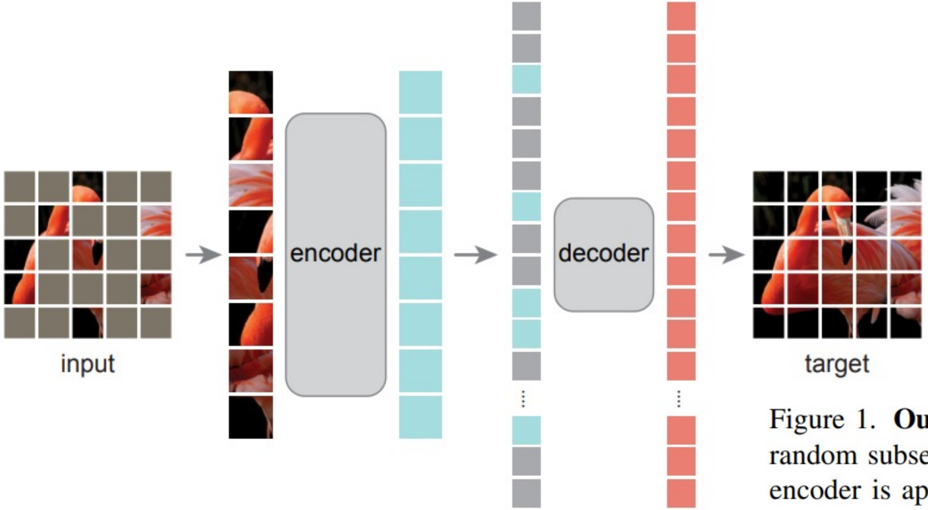
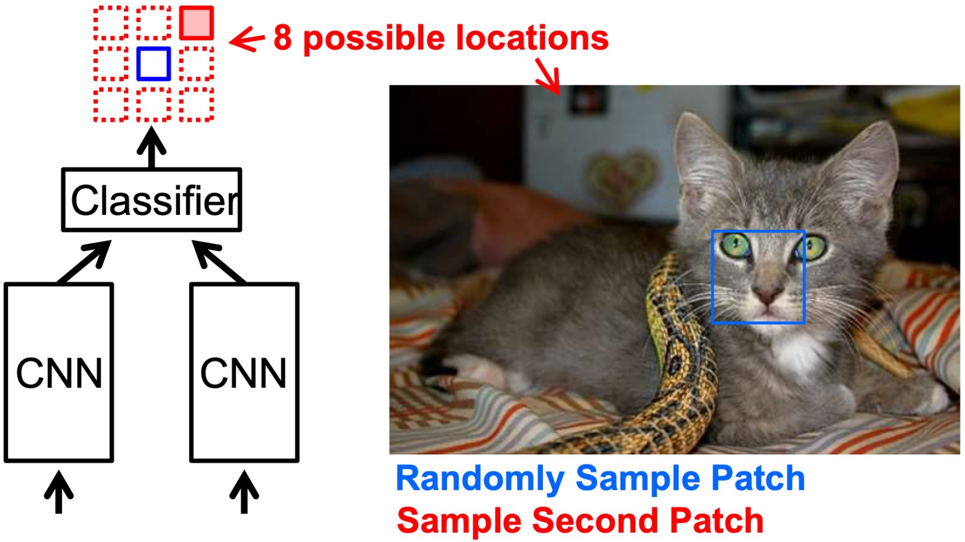


Figure 1. **Our MAE architecture**. During pre-training, a large random subset of image patches (*e.g.*, 75%) is masked out. The encoder is applied to the small subset of *visible patches*. Mask tokens are introduced *after* the encoder, and the full set of encoded patches and mask tokens is processed by a small decoder that reconstructs the original image in pixels. After pre-training, the decoder is discarded and the encoder is applied to uncorrupted images (full sets of patches) for recognition tasks.

[He et al., 2021: Masked Autoencoders Are Scalable Vision Learners]

SSL from Images, EX (II): relative positioning

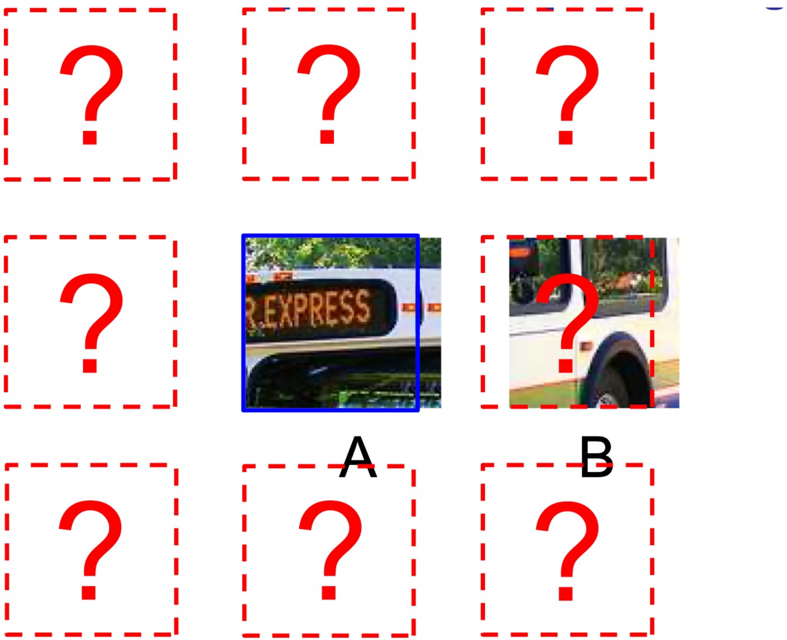
Train network to predict relative position of two regions in the same image



Unsupervised visual representation learning by context prediction,
Carl Doersch, Abhinav Gupta, Alexei A. Efros, ICCV 2015

[Courtesy: Zisserman "Self-supervised Learning"]

SSL from Images, EX (II): relative positioning



Unsupervised visual representation learning by context prediction,
Carl Doersch, Abhinav Gupta, Alexei A. Efros, ICCV 2015

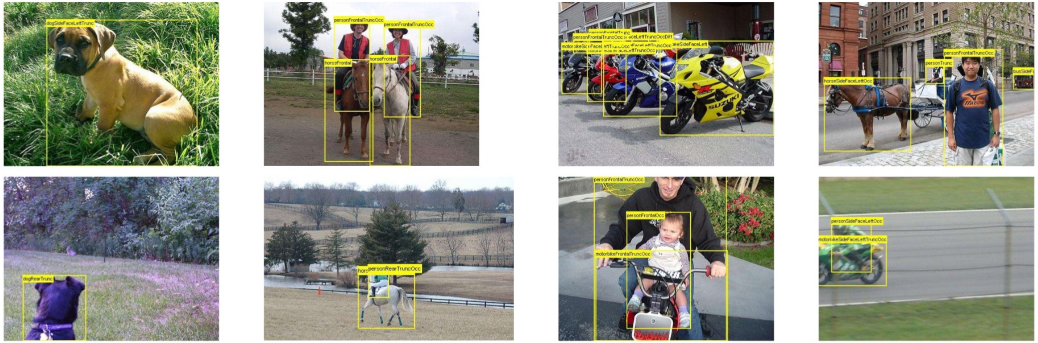
[Courtesy: Zisserman "Self-supervised Learning"]

SSL from Images, EX (II): relative positioning

Evaluation: PASCAL VOC Detection

- 20 object classes (car, bicycle, person, horse …)
- Predict the bounding boxes of all objects of a given class in an image (if any)

Dog Horse Motorbike Person



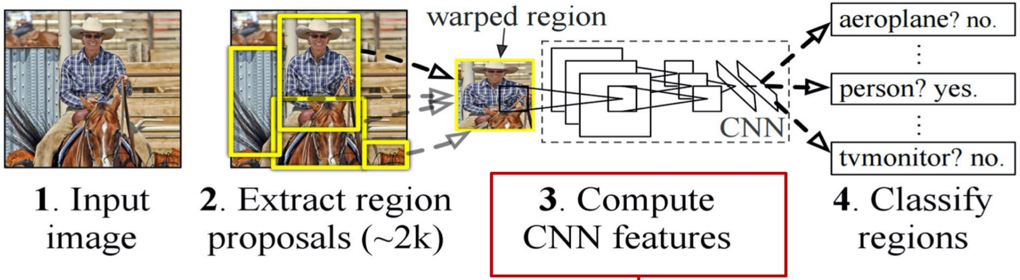
[Courtesy: Zisserman “Self-supervised Learning”]

SSL from Images, EX (II): relative positioning

Evaluation: PASCAL VOC Detection

- Pre-train CNN using self-supervision (no labels)
- Train CNN for detection in R-CNN object category detection pipeline

R-CNN



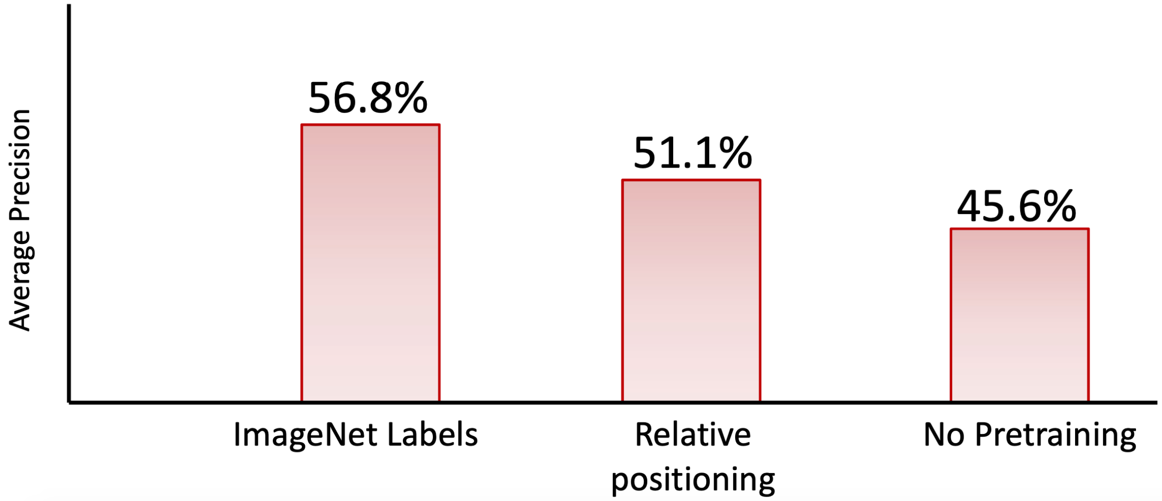
Pre-train on relative-position task, w/o labels

[Girshick et al. 2014]

[Courtesy: Zisserman "Self-supervised Learning"]

SSL from Images, EX (II): relative positioning

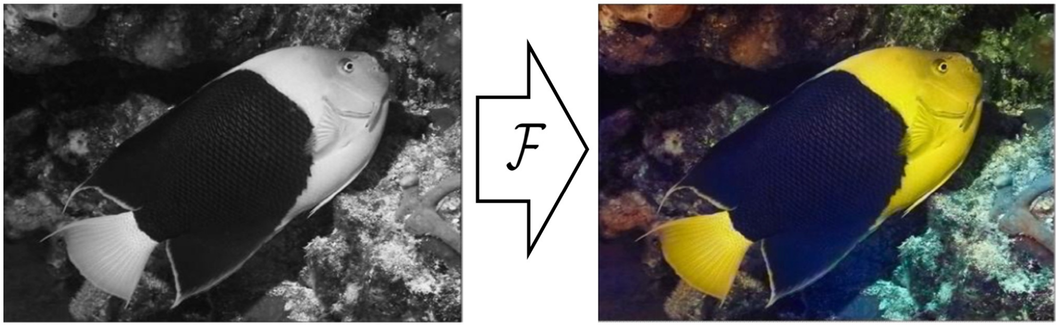
Evaluation: PASCAL VOC Detection



[Courtesy: Zisserman "Self-supervised Learning"]

SSL from Images, EX (III): colorization

Train network to predict pixel colour from a monochrome input

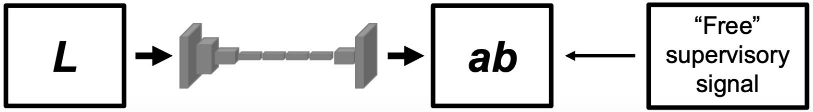


Grayscale image: *L* channel

Concatenate (L,ab)

$\mathbf{X} \in \mathbb{R}^{H \times W \times 1}$

$(\mathbf{X}, \widehat{\mathbf{Y}})$



[Courtesy: Zisserman "Self-supervised Learning"]

Colorful Image Colorization, Zhang et al., ECCV 2016

SSL from Images, EX (III): colorization

Train network to predict pixel colour from a monochrome input

[Courtesy: Zisserman “Self-supervised Learning”]

Colorful Image Colorization, Zhang et al., ECCV 2016

SSL from Images, EX (IV): exemplar networks

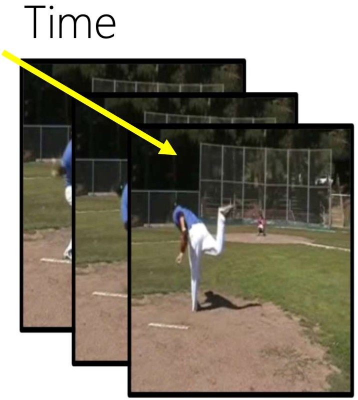
- Exemplar Networks (Dosovitskiy et al., 2014)
- Perturb/distort image patches, e.g. by cropping and affine transformations
- Train to classify these exemplars as same class

[Courtesy: Zisserman "Self-supervised Learning"]

SSL from Videos

Three example tasks:

- Video sequence order
  - Sequential Verification: Is this a valid sequence?



"Sequence" of data

[Courtesy: Zisserman "Self-supervised Learning"]

Wei et al., 2018 Arrow of Time

SSL from Videos

Three example tasks:

- Video sequence order
  - Sequential Verification: Is this a valid sequence?
- Video direction
  - Predict if video playing forwards or backwards

[Courtesy: Zisserman “Self-supervised Learning”]

Wei et al., 2018 Arrow of Time

SSL from Videos

Three example tasks:

- Video sequence order
  - Sequential Verification: Is this a valid sequence?
- Video direction
  - Predict if video playing forwards or backwards
- Video tracking
  - Given a color video, colorize all frames of a gray scale version using a reference frame

[Courtesy: Zisserman “Self-supervised Learning”]

Vondric et al., 2018

Key Takeaways

- Self supervision learning
  - Predicting any part of the observations given any available information
  - The prediction task forces models to learn semantic representations
  - Massive/unlimited data supervisions
- SSL for text:
  - Language models: next word prediction
  - BERT text representations: masked language model (MLM)
- SSL for images/videos:
  - Various ways of defining the prediction task

Questions?