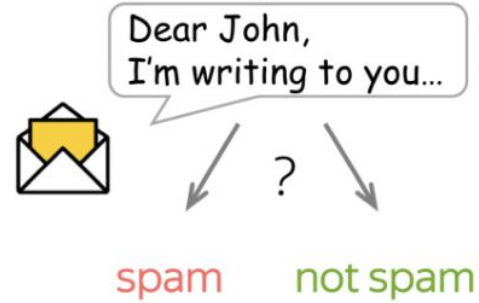


Lecture 4:

Transfer Learning in NLP

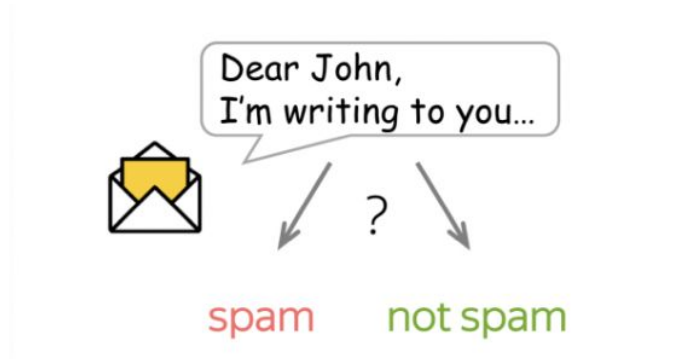
Transfer Learning

Let's say we need to train a text classification model



Transfer Learning

Let's say we need to train a text classification model

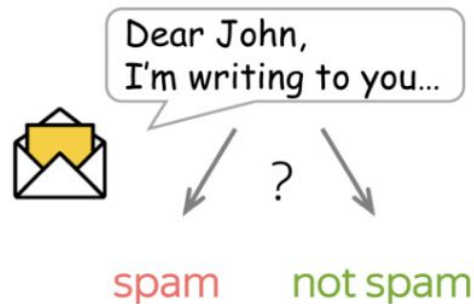


To train from scratch, we need

- Labeled samples
- Pre-extracted features

Transfer Learning

Let's say we need to train a text classification model



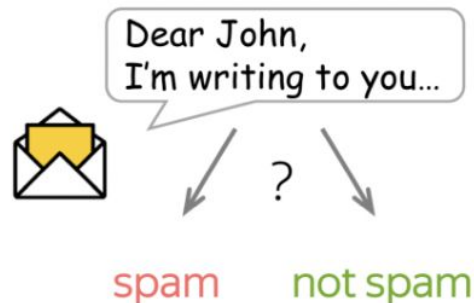
To train from scratch, we need

- Labeled samples
- Pre-extracted features

Expensive to obtain

Transfer Learning

Let's say we need to train a text classification model

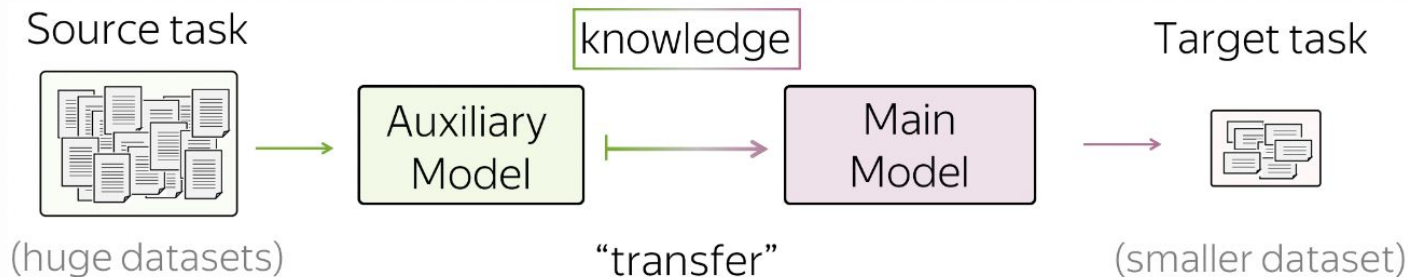


To train from scratch, we need

- Labeled samples
- Pre-extracted features

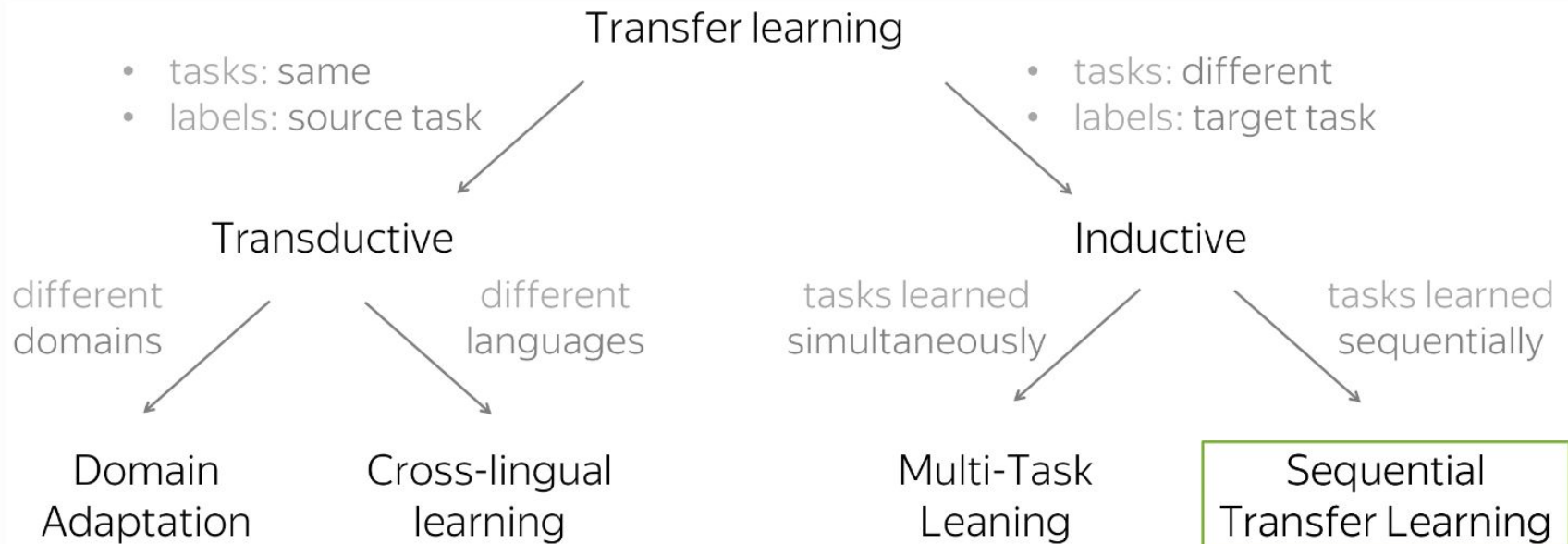
Expensive to obtain

Transfer Learning

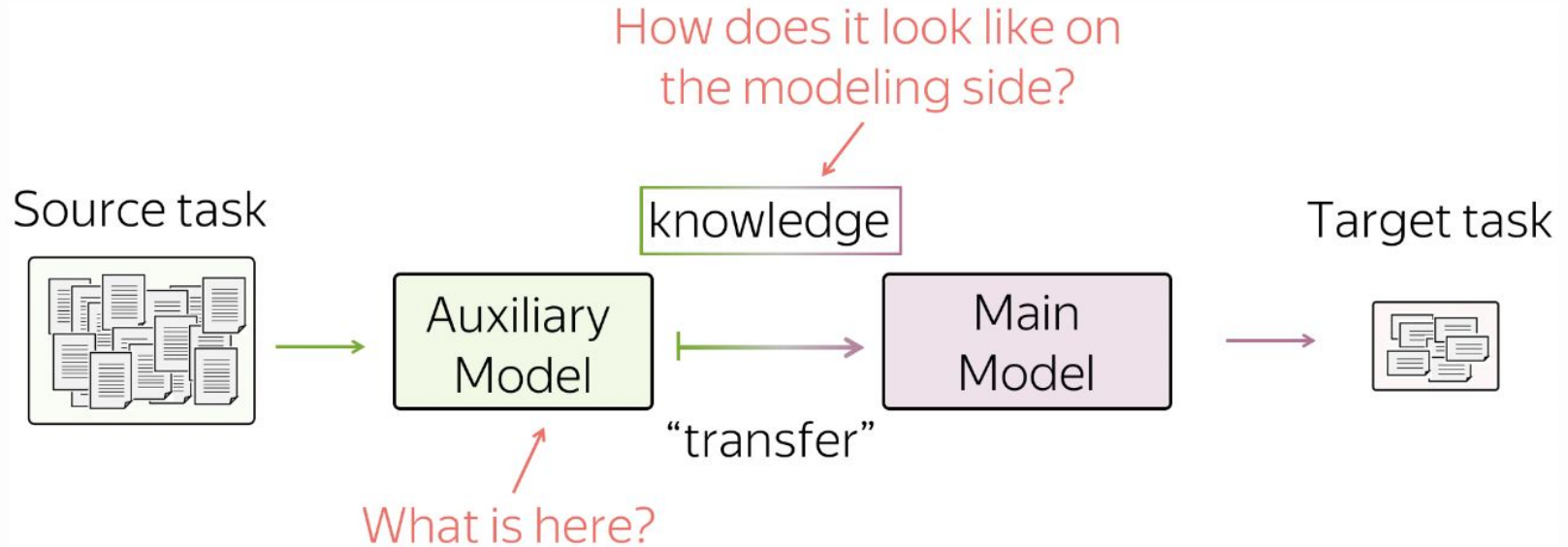


Transfer Learning is a technique of exploiting the properties and data distribution of a given (task, dataset) pair on different tasks and datasets of interest

Transfer Learning Taxonomy

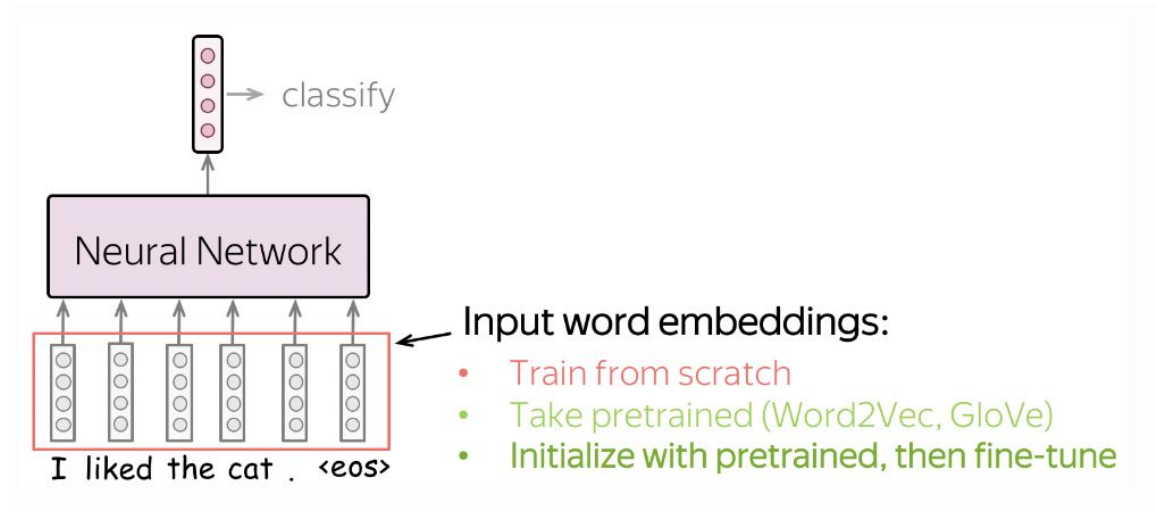


Sequential Transfer Learning



Transfer Learning via Word Embeddings

- Let's look at embedding layer in classification pipeline



Transfer Learning via Word Embeddings

- Which data distribution was embedding layer attuned to?

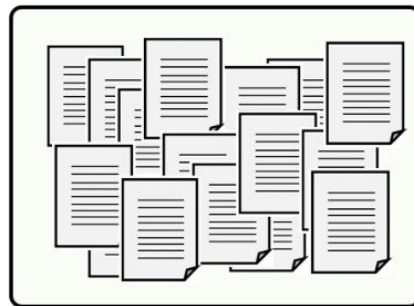
Embedding layer from scratch
(trained jointly for cls)



Training data for text classification (labeled)

- Not huge, or not diverse, or both
- Domain: task-specific

Embedding layer == word2vec



Training data for word embeddings (unlabeled)

- Huge diverse corpus (e.g., Wikipedia)
- Domain: general

Transfer Learning via Word Embeddings

- Train from scratch

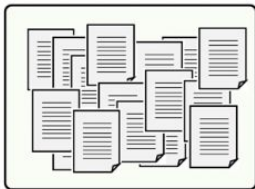
What they will know:



May be not enough to
learn relationships
between words

- Take pretrained
(Word2Vec, GloVe)

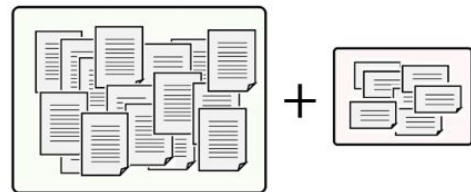
What they will know:



Know relationships between words,
but are **not** specific to the task

- Initialize with pretrained,
then fine-tune

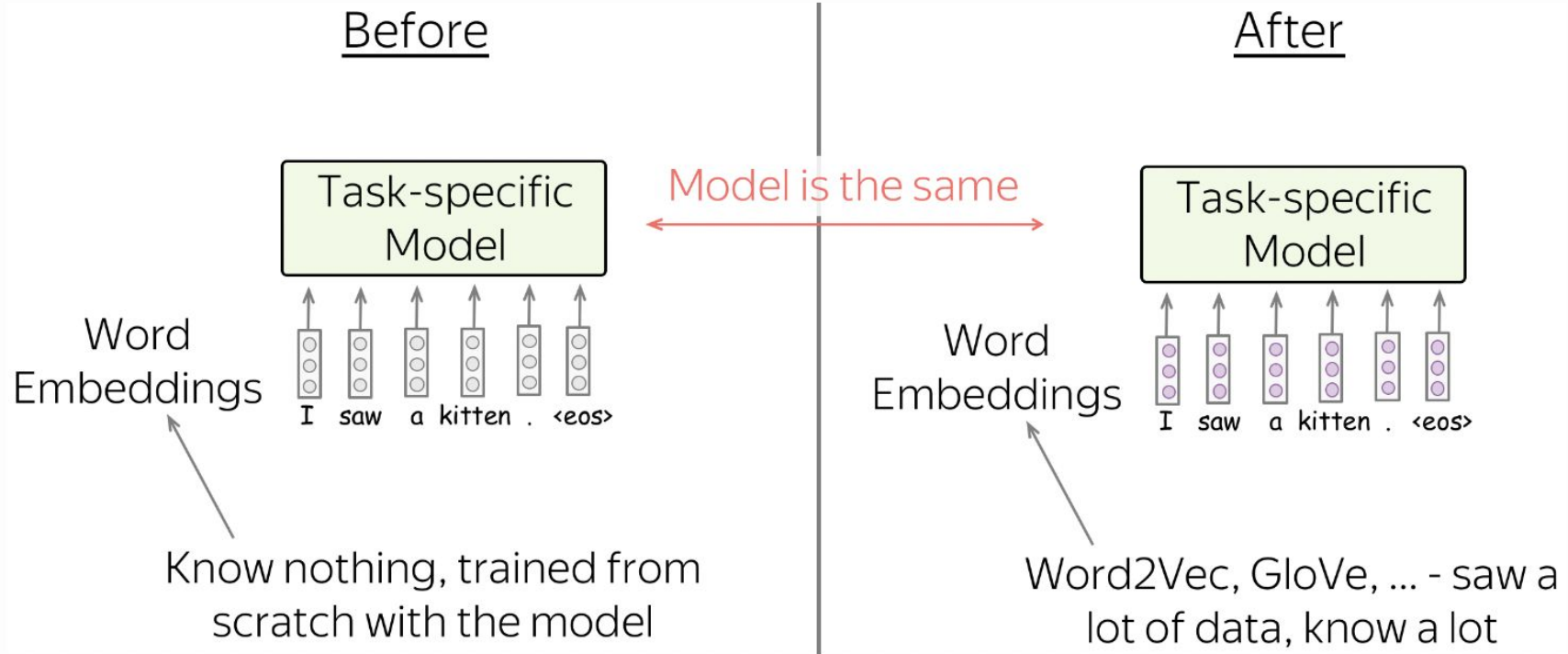
What they will know:



Know relationships between
words and adapted for the task

“Transfer” knowledge from a huge unlabeled corpus to your task-specific model

Transfer Learning via Word Embeddings

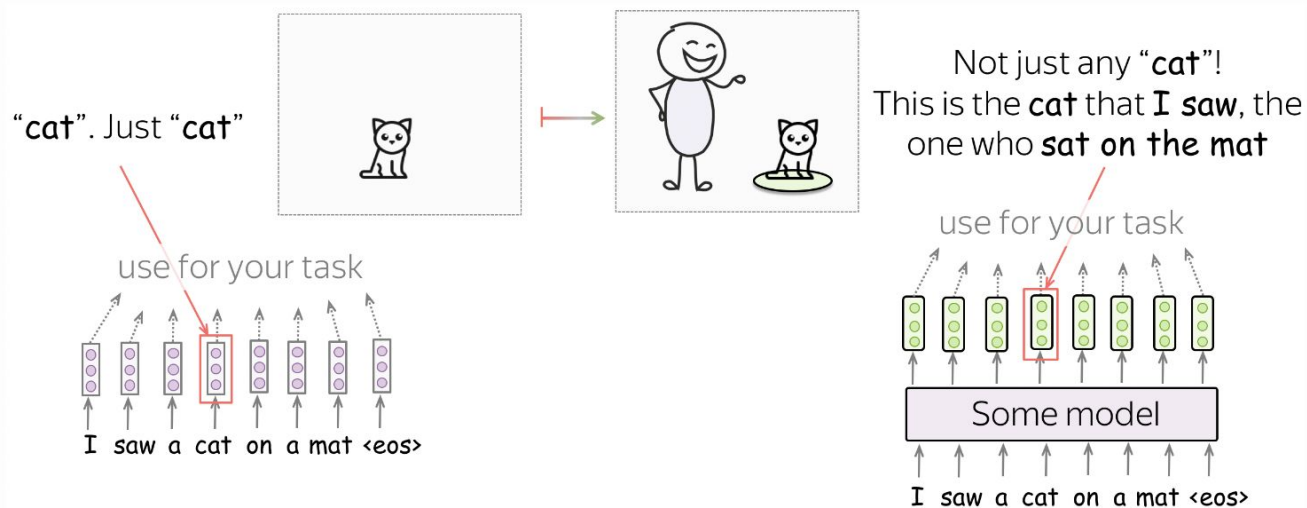


Transfer Learning via Contextualized Representations

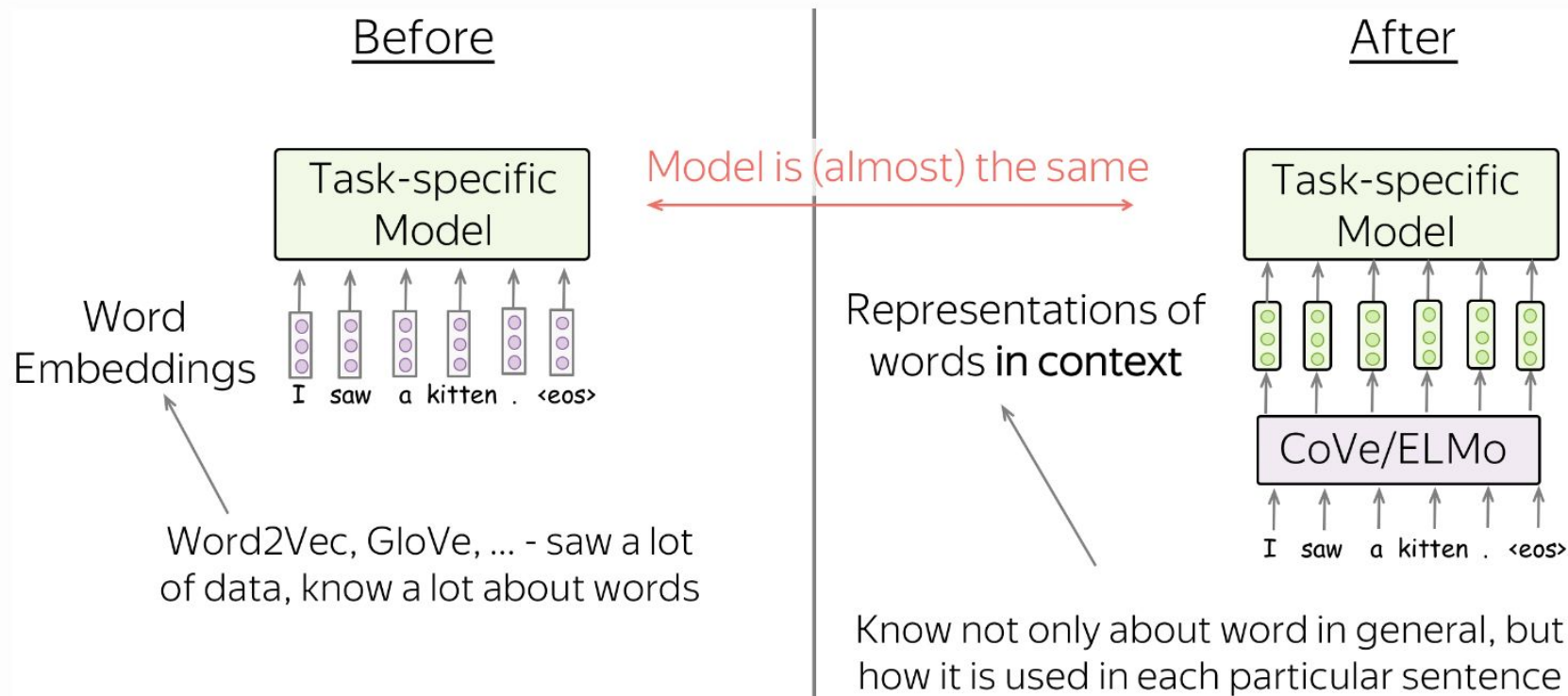
- Word vectors do not model any intra-token relationships

Transfer Learning via Contextualized Representations

- Word vectors do not model any intra-token relationships
- The same way as words, we can learn to encode words along with the context they are used in



Transfer Learning via Contextualized Representations



Representation Learning (CoVe, ELMO)

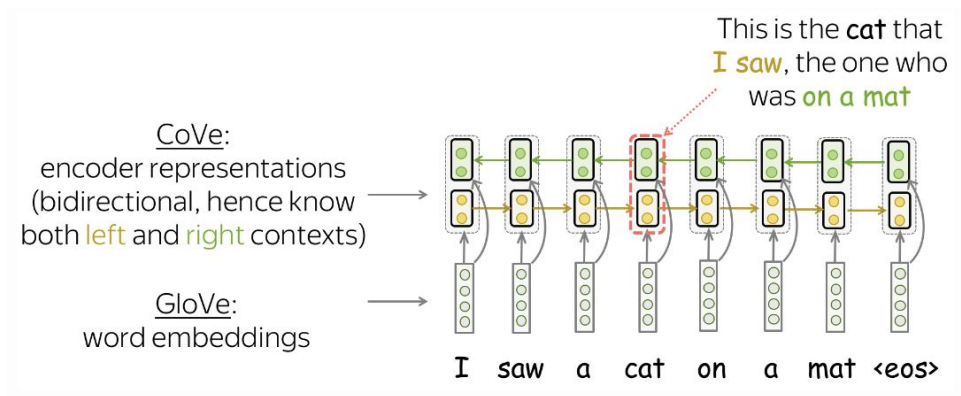
CoVe

CoVe (Contextualized Word Vectors Learned in Translation)

Key Idea: translation of sentence requires modeling complex token dependencies
and NMT model learns to “understand” sentence

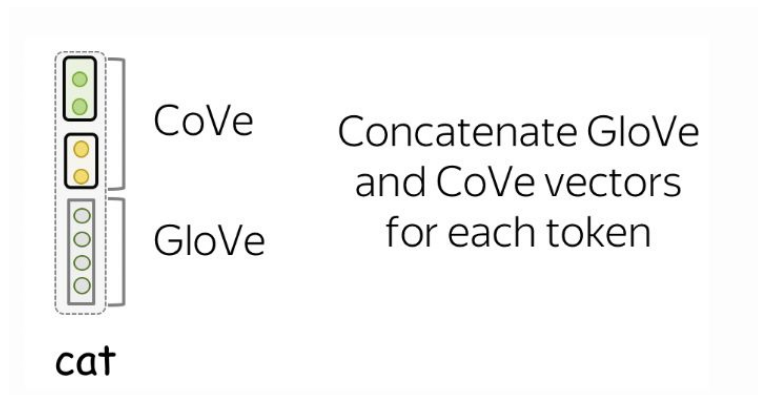
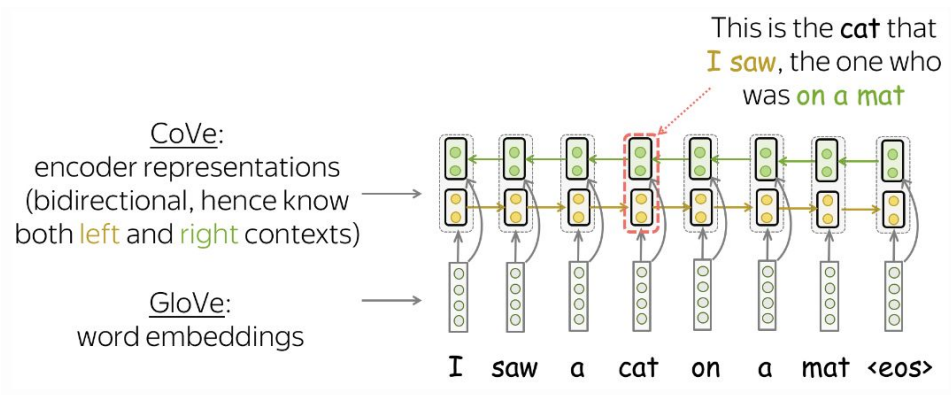
CoVe

- Train NMT encoder-decoder system
- Take pre-trained encoder as feature extractor
- CoVe = encoder outputs for a given sequence



CoVe

- Train NMT encoder-decoder system
- Take pre-trained encoder as feature extractor
- CoVe = encoder outputs for a given sequence
- For downstream tasks: CoVe + GloVe embeddings



ELMO

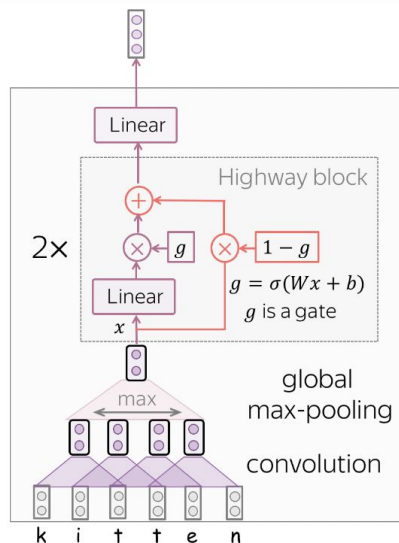
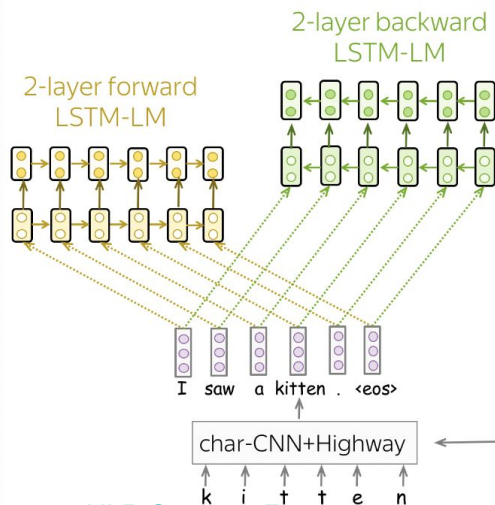
ELMO (Embeddings From Language Models)

Key Idea: similar to CoVe, but instead of NMT use LM pretraining objective

ELMO

ELMO (Embeddings From Language Models)

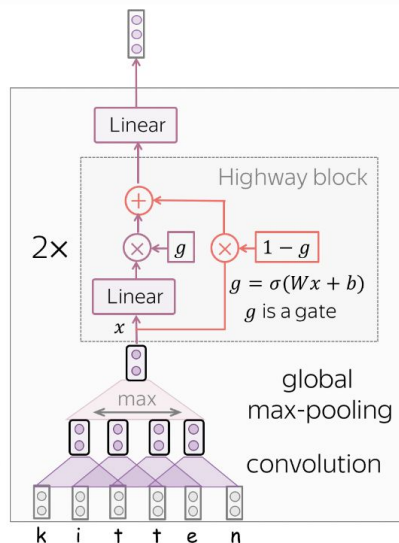
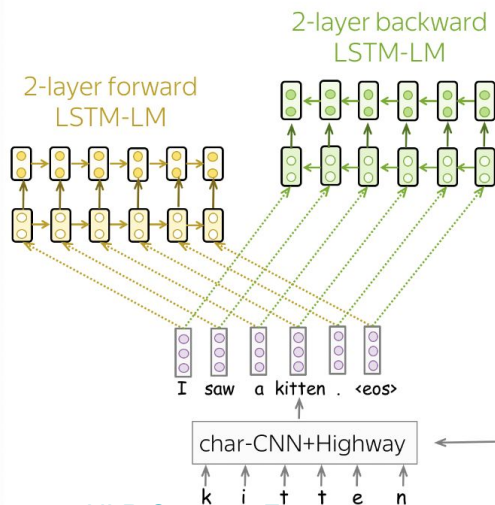
Key Idea: similar to CoVe, but instead of NMT use LM pretraining objective



ELMO

ELMO (Embeddings From Language Models)

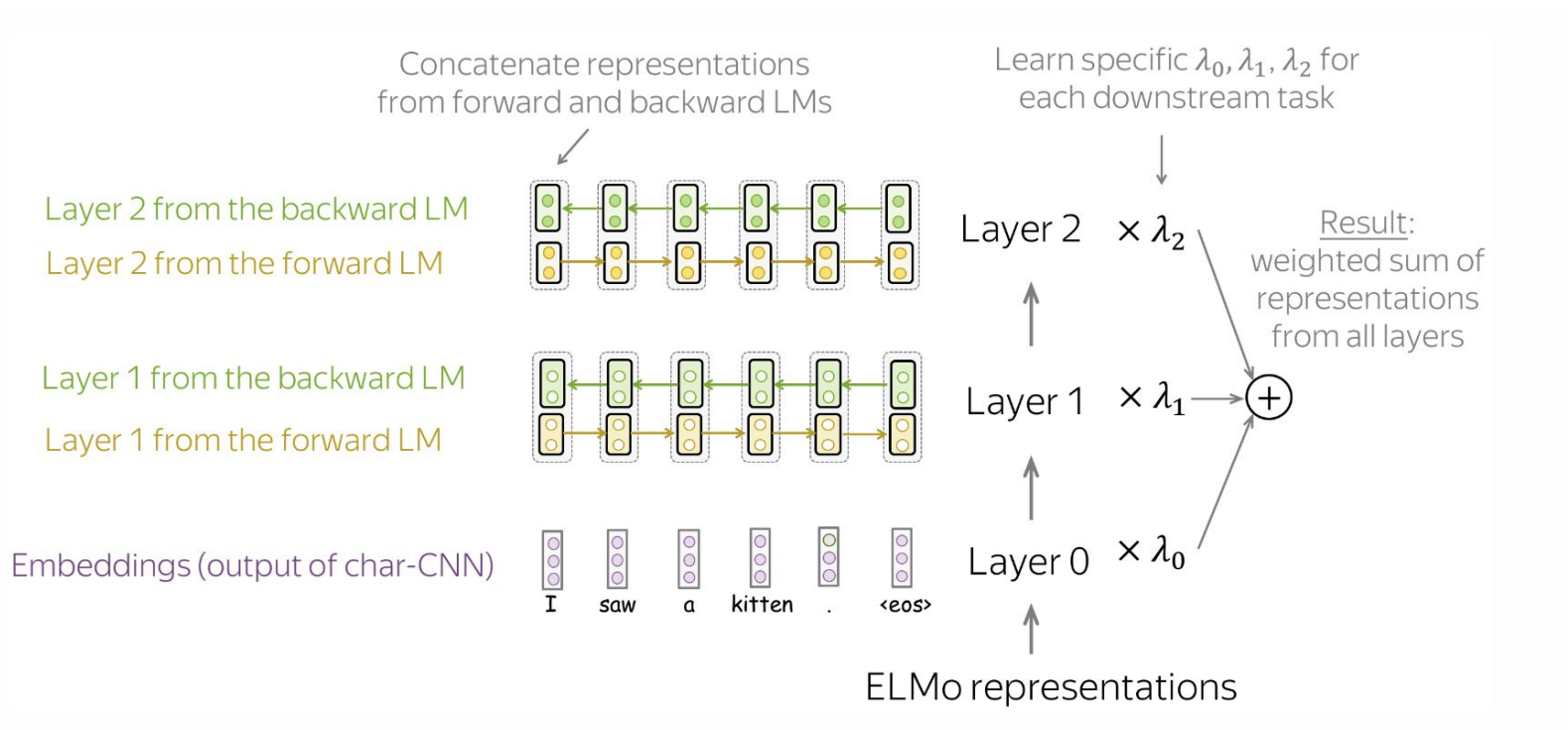
Key Idea: similar to CoVe, but instead of NMT use LM pretraining objective



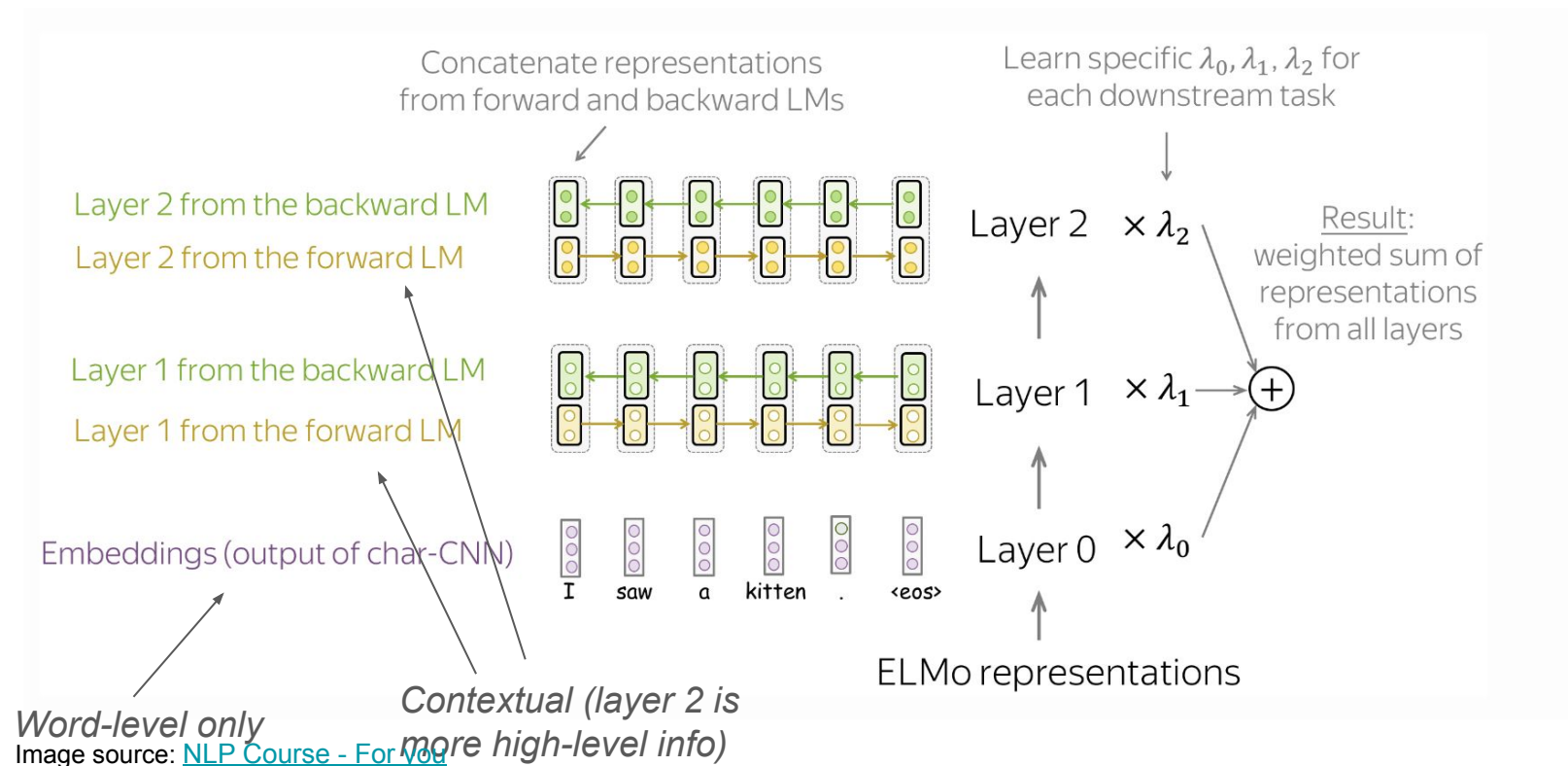
Initialization

*Character-level CNN
(robust handling of OOV
words)*

ELMO: Feature Extraction



ELMO: Feature Extraction



Transfer Learning via Pretrained Models

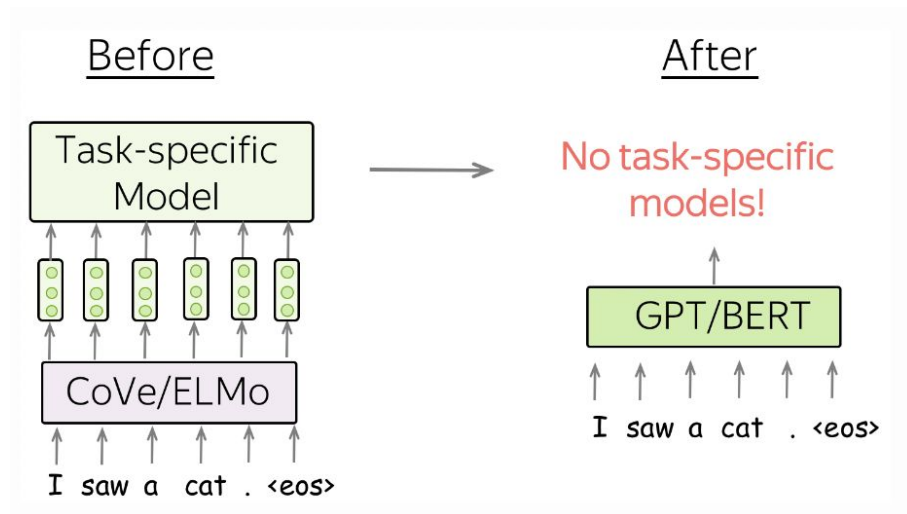
Transfer Learning

Contextualized Representations

- Replace word embedding layer
- Rest of the pipeline should be trained independently for each task

Pretrained Models

- Replace full task-specific model
- Minimal adaptation or usage “as is”



Pretrained Models From Language Modeling

Language Modeling is a fertile task for representation learning

- Semi-supervised (labels are implicitly given in data)
- Ubiquitous datasets (web crawl, literature, etc.)
- Complicated and very generic

GPT

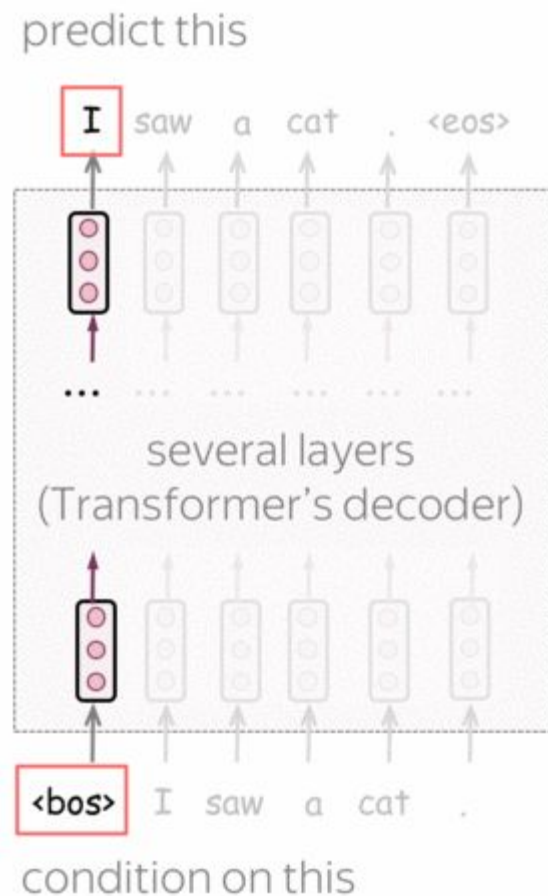
GPT (Generative Pre-Training for Language Understanding)

Key Idea: autoregressive LM with transformer decoder

$$L_{xent} = - \sum_{t=1}^n \log(p(y_t | y_{<t})).$$

GPT

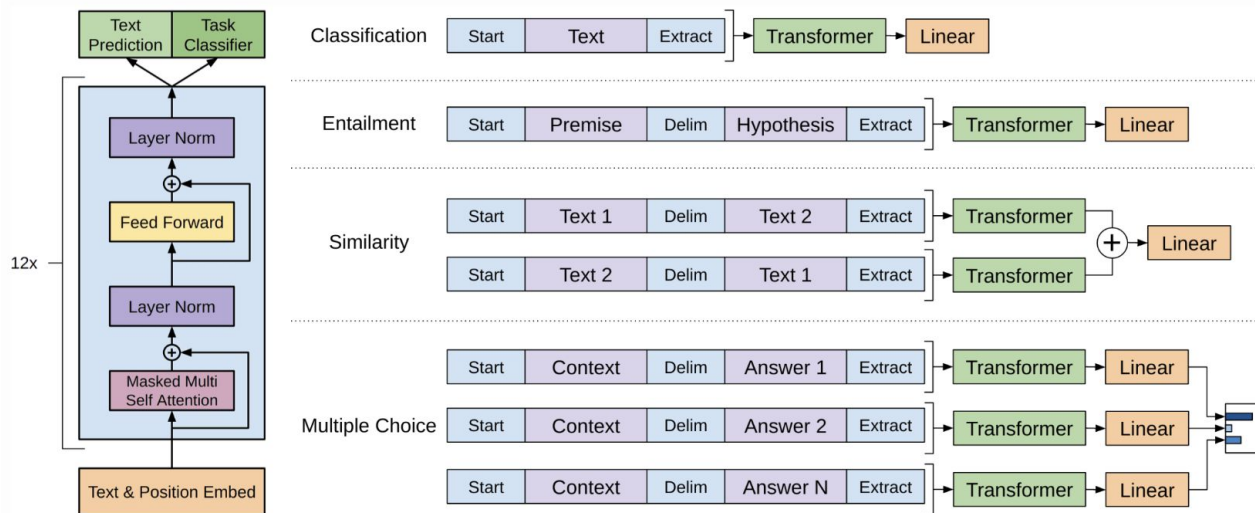
- Decoder-only (triangle attention mask)
- Autoregressively predict next token conditioned on left context
- Train on unsupervised corpora with supervised xent loss (self-supervised learning)



GPT Finetuning (GPT-1)

- Finetune LM parameters using a combination of two losses

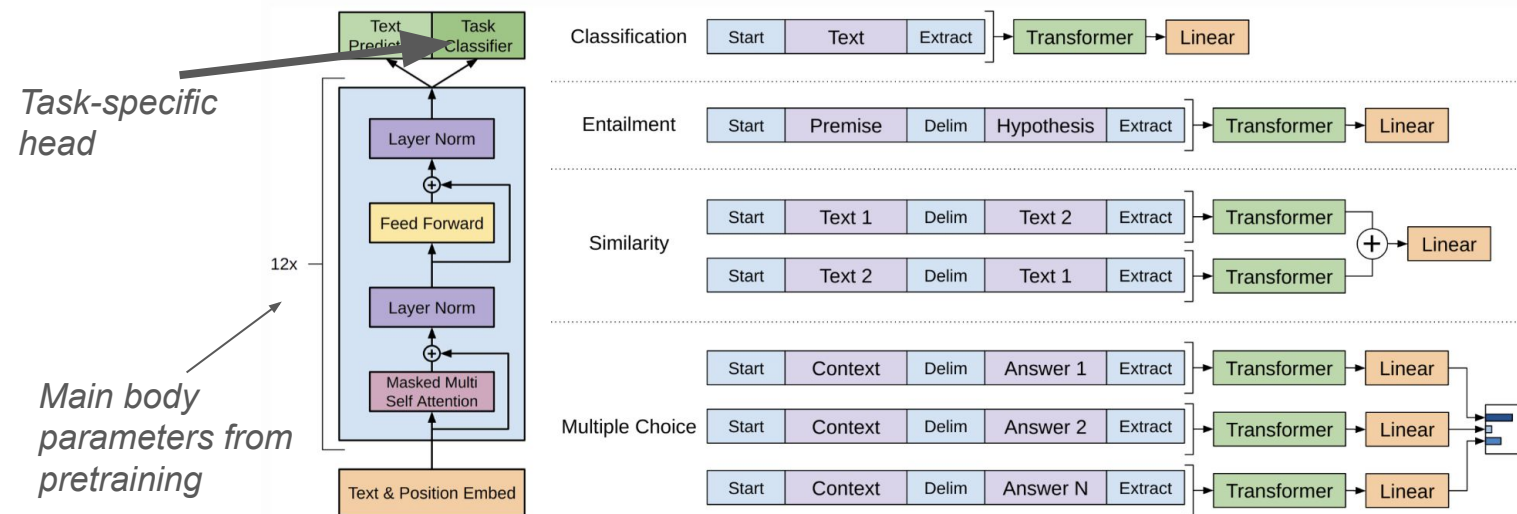
$$L = L_{xent} + \lambda \cdot L_{task}.$$



GPT Finetuning (GPT-1)

- Finetune LM parameters using a combination of two losses

$$L = L_{\text{rent}} + \lambda \cdot L_{\text{task}}.$$



BERT

BERT

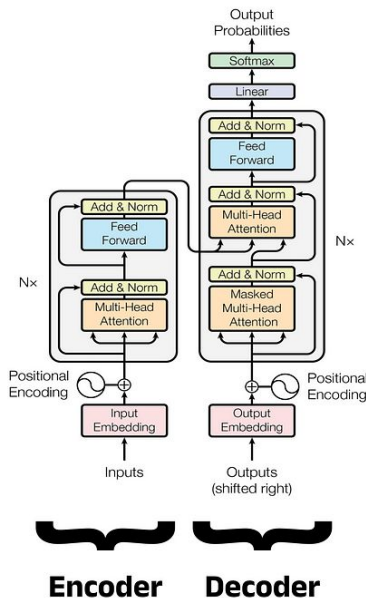
BERT (Bidirectional Encoder Pre-training for Transformers)

Key Idea: language modeling task + transformer encoder

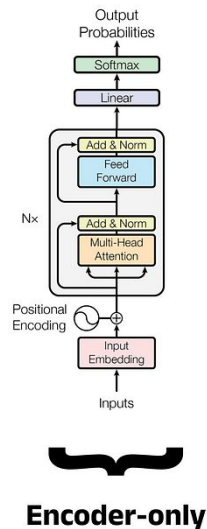
BERT

- Encoder-only
- Bidirectional Encoder Representations from Transformers (MLM task)
- Based on Transformer architecture
- Bidirectional context encoding
- Pretrained on Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) tasks

Transformer

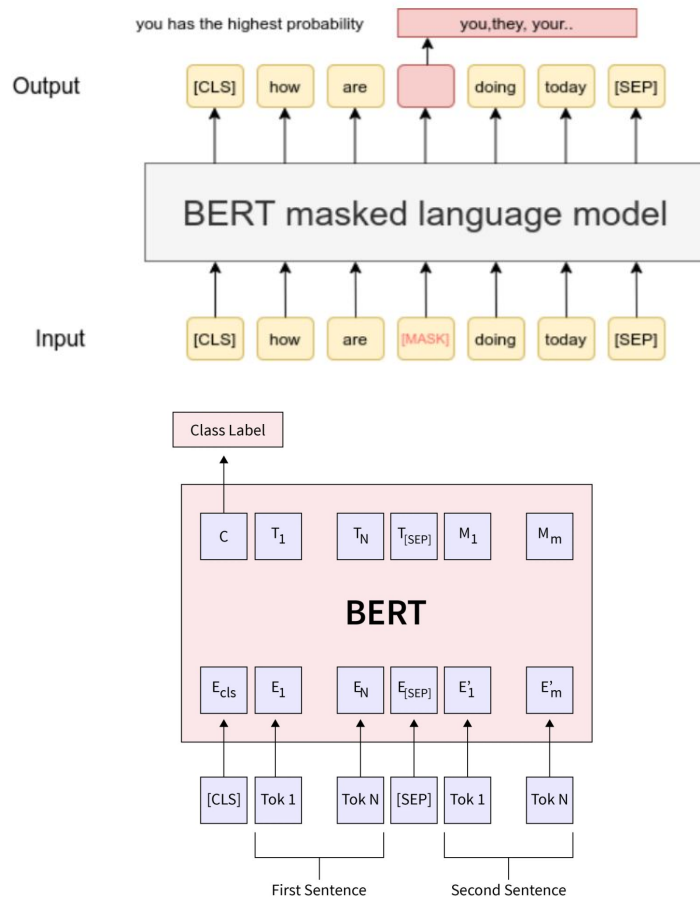


BERT*



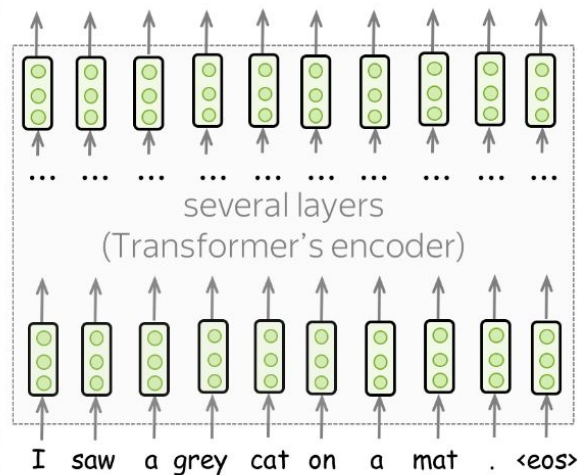
BERT. Pretrain Tasks.

- **Masked Language Model (MLM)**
 - 15% of tokens are randomly masked
 - 80% replaced with [MASK]
 - 10% replaced with a random word
 - 10% remain unchanged
- **Next Sentence Prediction (NSP)**
 - Model learns to predict if sentence B logically follows sentence A



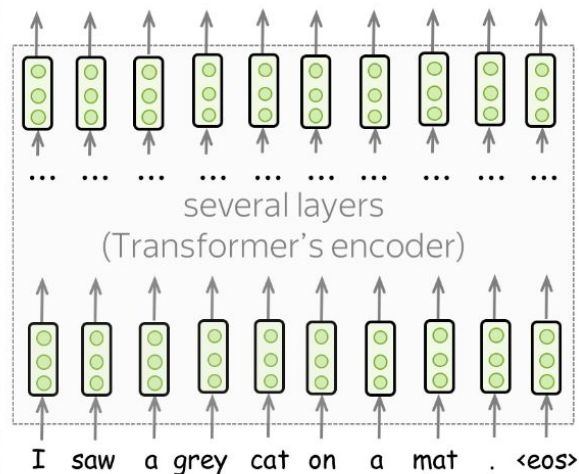
BERT

Q.: how to do LM using transformer encoder (no triangle mask) ?



BERT

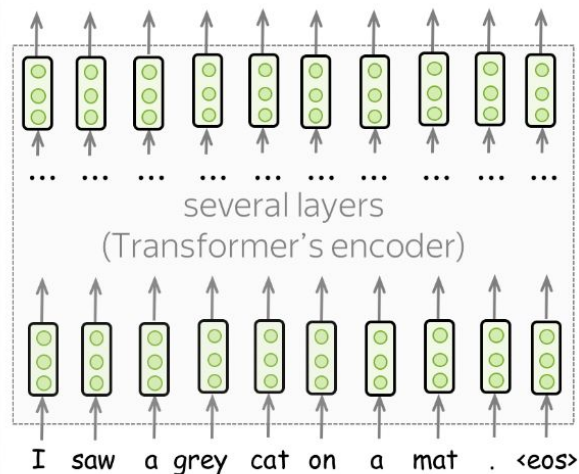
Q.: how to do LM using transformer encoder (no triangle mask) ?



- Standard L2r objective can not be used due to “lookahead”
- Let’s hide some tokens and train the model to predict masked positions

BERT

Q.: how to do LM using transformer encoder (no triangle mask) ?



- Standard L2r objective can not be used due to “lookahead”
- Let’s hide some tokens and train the model to predict masked positions

- Replace with special [MASK token]
- Classification head over final layer embeddings

BERT pretraining objective

First part: Masked Language Modeling (MLM)

Second part: model pairwise sentence dependencies

- NSP (Next Sentence Prediction)
- Whether or not two sentences directly follow each other
- [CLS] token to encode “full sentence meaning”

BERT pretraining objective

First part: Masked Language Modeling (MLM)

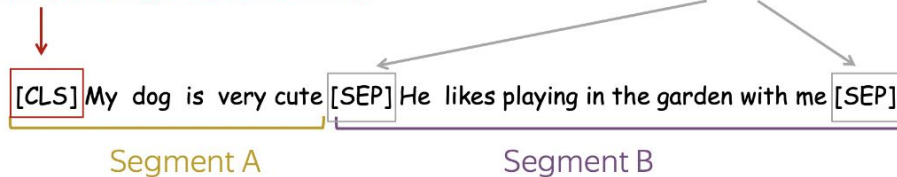
Second part: model pairwise sentence dependencies

- NSP (Next Sentence Prediction)
- Whether or not two sentences directly follow each other
- [CLS] token to encode “full sentence meaning”

[CLS]: Special token

- Training time: predict if sentences are consecutive or not (Next Sentence Prediction /NSP objective)
- Test time: downstream tasks (e.g., classification)

[SEP]: Special token-separator



BERT Pretraining (NSP)

Training time: predict if sentences are consecutive (NSP objective)

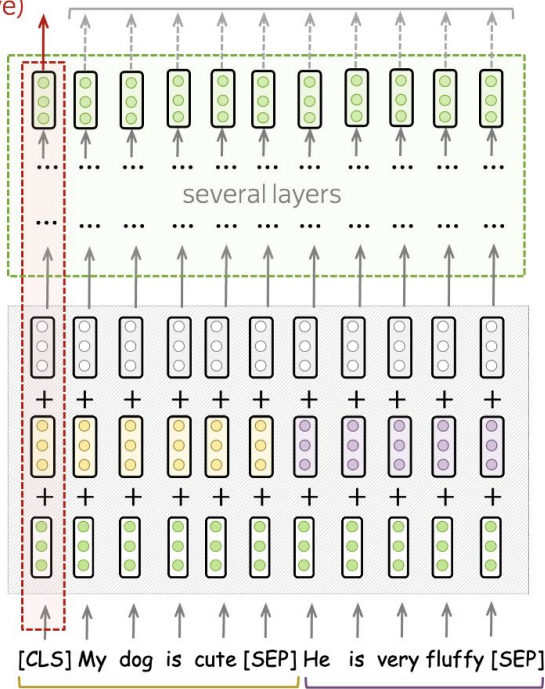
Test time: classification

Training time: MLM objective

Model
(Transformer
encoder)

Input

Training on pairs of sentences: either consecutive or random (50%/50%)



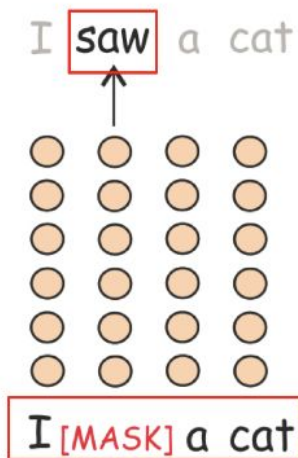
positions
0 1 2 3 4 ...

segments
A A A A A A B B B B B

tokens
[CLS] My dog is ...

BERT Pretraining (MLM)

- Target: current token (the true one)
- Prediction: $P(* | \mathbf{I} [\text{MASK}] \text{ a cat})$



sees the whole text, but
something is corrupted

BERT Finetuning

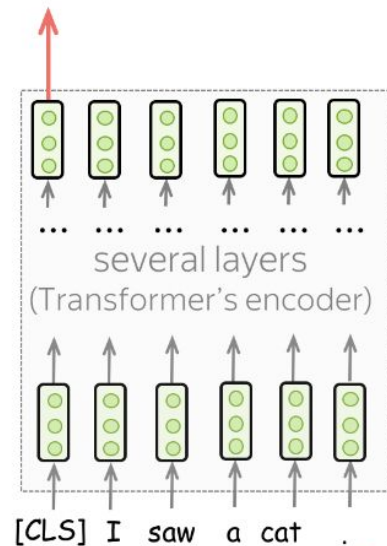
Sentence Classification

BERT Finetuning

Sentence Classification

- Input: [CLS] + Sent
- [CLS] embedding as a feature

class label



No second sentence!

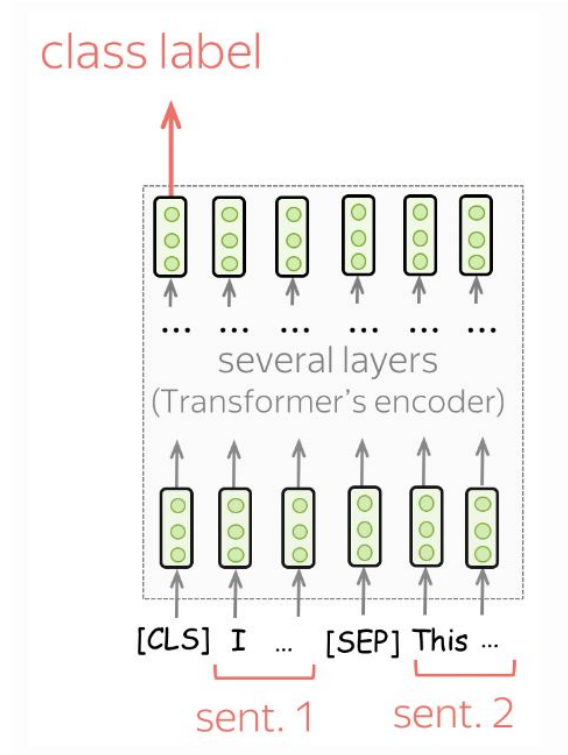
BERT Finetuning

Sentence Pair Classification

BERT Finetuning

Sentence Pair Classification

- Input: [CLS] + Sent1 + [SEP] + Sent2
- [CLS] embedding as feature



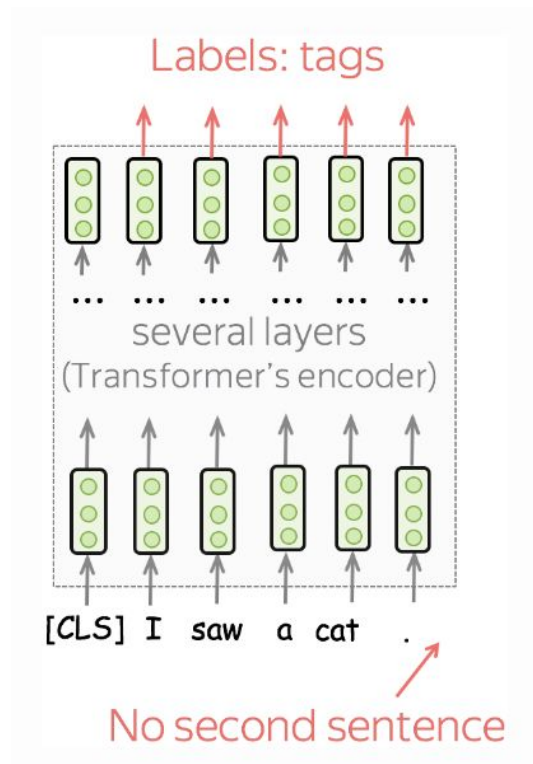
BERT Finetuning

Sentence Tagging (token classification)

BERT Finetuning

Sentence Tagging (token classification)

- Input: [CLS] + sent
- Each token's feature is its embedding



Thanks for attention!

*This lecture heavily uses plots and images from amazing [NLP Course - For you](#) by Lena Voita.
Check it out if you wish to dig deeper!*