NLP EXPO

Data

1. Text/Audio

   Natural Language Processing

2. Audio

   Speech Recognition

3. image/video

   Computer Vision

Natural Language Processing

What is NLP?

1. NLP is a subfield of **computer science, information engineering, and artificial intelligence**
2. concerned with the interactions between computers and human languages
3. in particular **how to program computers to process and analyze** large amounts of **natural language data.**

how to process and analyze natural language data?

Answer: We Need **Embedding Vectors**

5 steps of AI/ML Lifecycle

1. Data processing
2. Feature Extraction
3. Apply ML Algorithms
4. Validation/Test Accuracy
5. Fine tuning

Which one playing major role here?

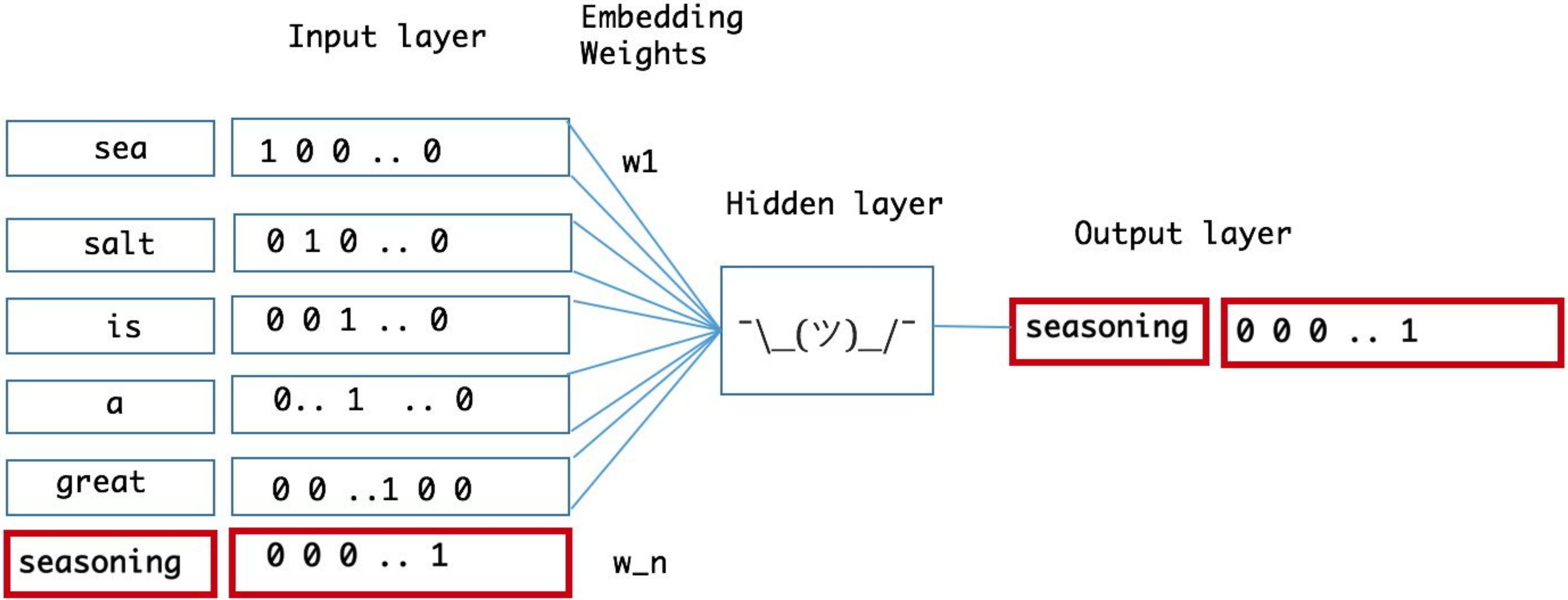
Answer: **Data Processing & Feature Extraction**

1. Well Cleaned data and
2. Efficient Feature Embeddings will produce best results forever

Let’s Go For Feature Embeddings

Embedding

1. An **embedding** is a relatively low-dimensional space into which you can translate high-dimensional vectors.
2. **Embeddings** make it easier to do **machine learning** on large inputs like sparse vectors representing words.
3. A **word embedding** that is learned jointly with a neural network model on a specific **natural language processing** task, such as language modeling or document classification.



Input layer

Embedding Weights

sea	1 0 0 .. 0
salt	0 1 0 .. 0
is	0 0 1 .. 0
a	0.. 1 .. 0
great	0 0 ..1 0 0
seasoning	0 0 0 .. 1

w1

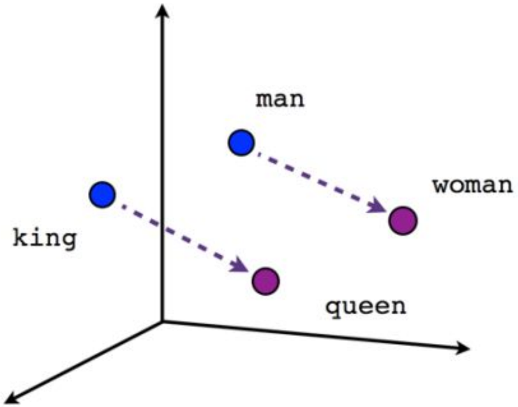
w_n

Hidden layer

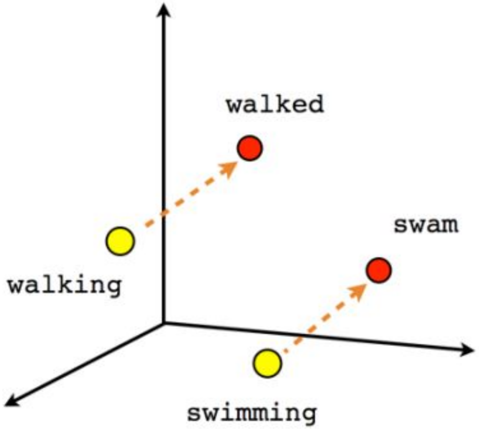
¯_(ツ)_/¯

Output layer

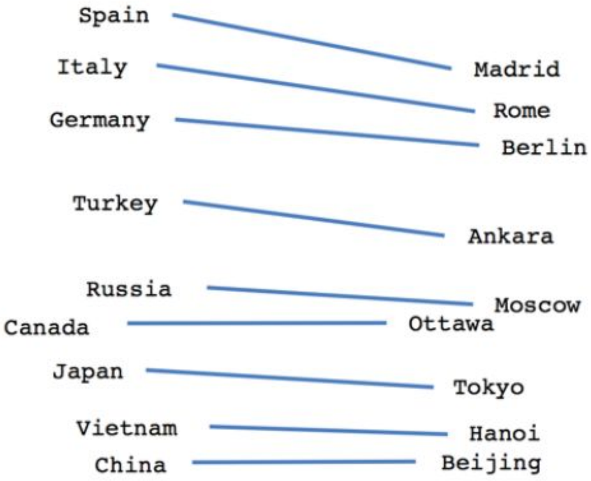
seasoning 0 0 0 .. 1



Male-Female



Verb tense



Country-Capital

Word Embeddings - Traditional

To use word embeddings, you have three primary options:

- Use pre-trained models that you can download online (easiest)
- Train custom models using your own data
- Resuming training(Continue the training)

Pre-trained Word Embeddings

1. Using a pre-trained model removes the need for you to spend time obtaining, cleaning, and processing (intensively) such large datasets.
2. Pre-trained models are also available in languages other than English, opening up multi-lingual opportunities for your applications.

Pre-trained Word Embeddings

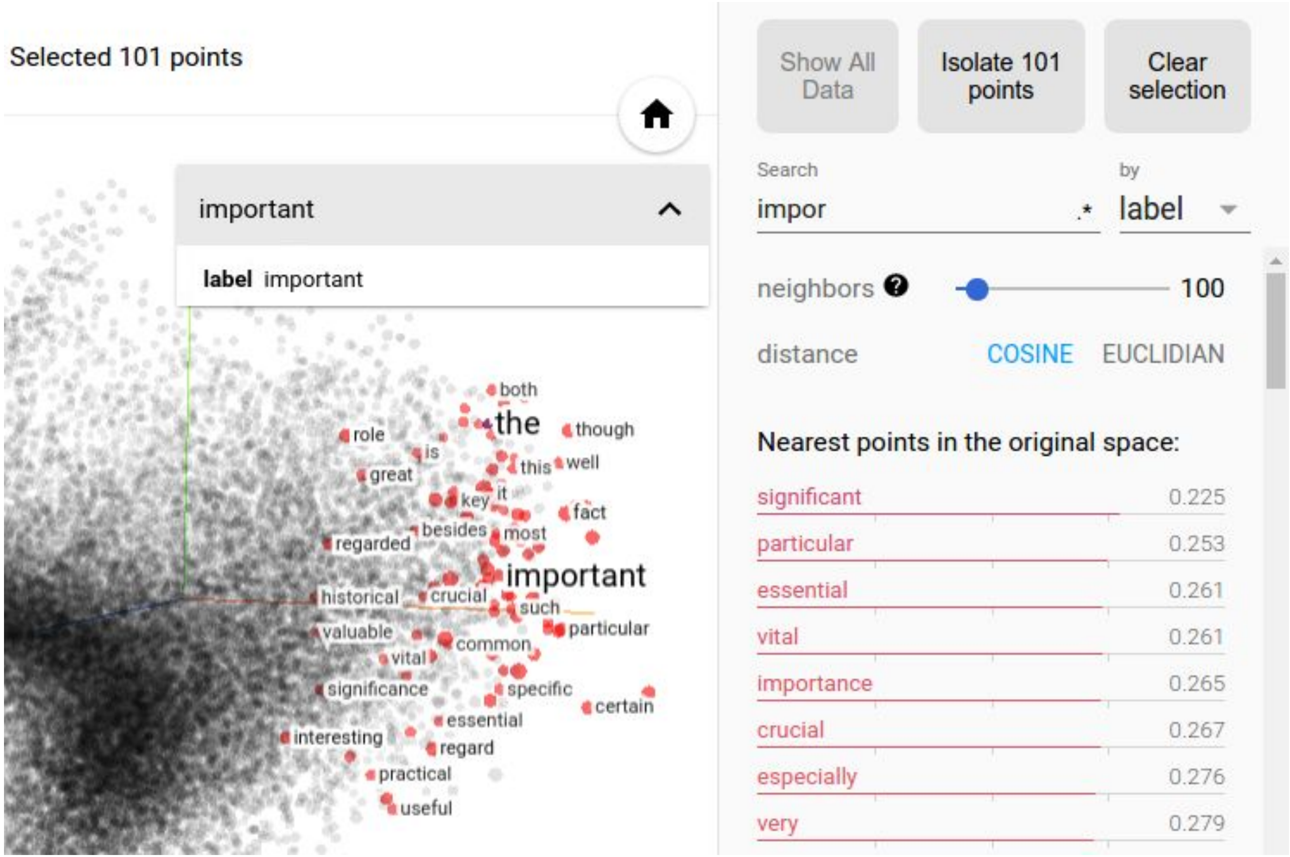
The disadvantage of pre-trained word embeddings

1. the words contained within may not capture the peculiarities of language in your specific application domain.
2. Wikipedia may not have great word exposure to particular aspects of legal doctrine or religious text, so if your application is **specific to a domain** like this, your results **may not be optimal** due to the generality of the downloaded model’s word embeddings.

Word Embeddings Visualization

Let’s try this:

https://projector.tensorflow.org/



Selected 101 points

important

label important

Search impor .* by label

neighbors 100

distance COSINE EUCLIDIAN

Nearest points in the original space:

significant	0.225
particular	0.253
essential	0.261
vital	0.261
importance	0.265
crucial	0.267
especially	0.276
very	0.279

What is Language Modelling?

1. A statistical language model is a **probability distribution over sequences of words.**
2. Given such a sequence, say of length m, it assigns a probability to the whole sequence.
3. The language model **provides context to distinguish between words** and phrases that sound similar.

language models have played a key role in traditional NLP tasks such as speech recognition, machine translation, or text summarization. ***training better language models improves*** *the underlying metrics of the downstream task (such as word error rate for speech recognition, or BLEU score for translation), which* ***makes the task of training better*** *LMs valuable by itself.*

Traditional Language Modelling Algorithms

1. Skip-gram
2. Continuous Bag Of Words

Loss Functions

3. Nagative Sampling
4. Hierarchy Softmax
5. Noise Contrastive Estimation
6. Cross Entropy

Open Source Libraries

Gensim

NLTK

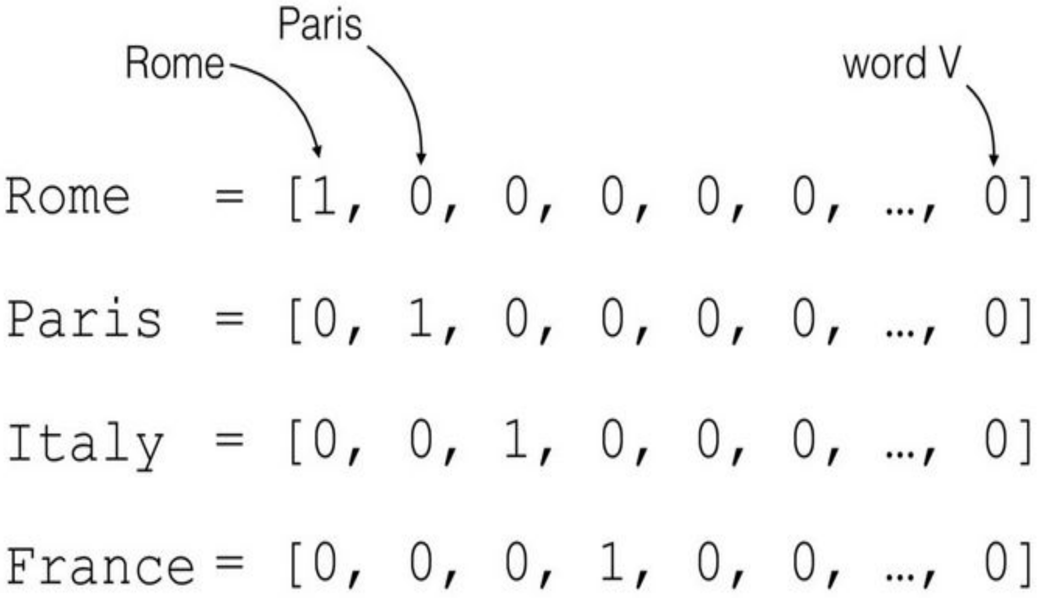
Spacy

Open Source Community

Word2Vec, BERT	- Google
GLoVe	- Stanford NLP
Fasttext	- Facebook AI Research
ELMo	- AllenNLP
GPT-2	- OpenAI

Onehot Encoding

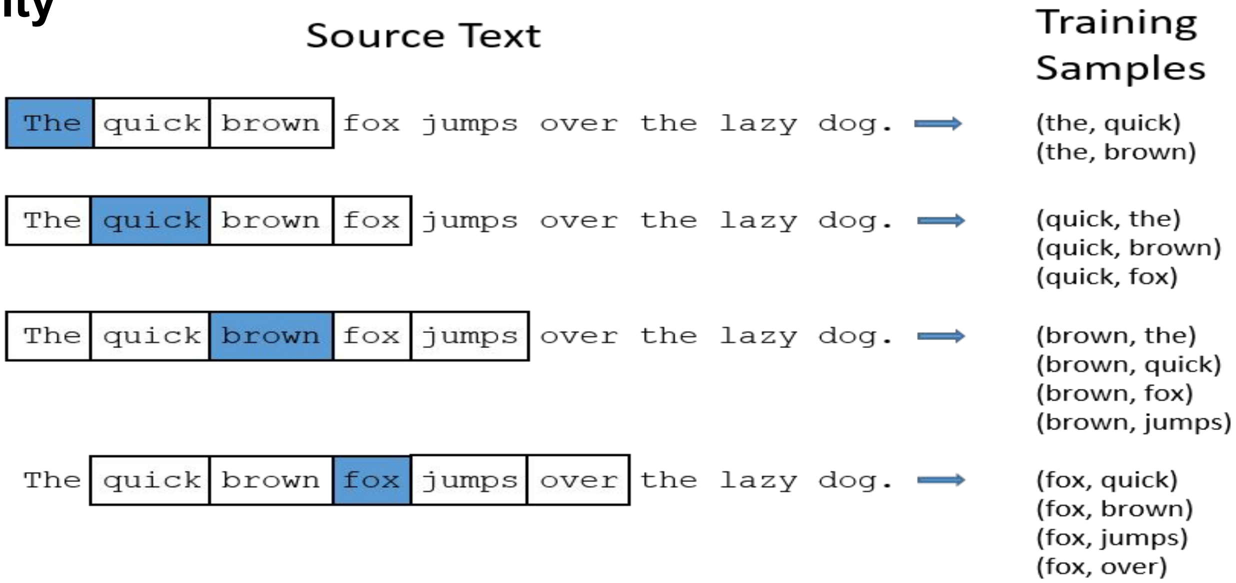
One hot encoding is a process by which **categorical variables** are converted into a form that could be provided to ML algorithms to **do a better job in prediction**.



```
Rome   = [1, 0, 0, 0, 0, 0, …, 0]
Paris  = [0, 1, 0, 0, 0, 0, …, 0]
Italy  = [0, 0, 1, 0, 0, 0, …, 0]
France = [0, 0, 0, 1, 0, 0, …, 0]
```

Skip-gram

find word representations that are useful for predicting the surrounding words in a sentence or a document. given a sequence of training words is to **maximize the average log probability**



Source Text

Training Samples

The quick brown fox jumps over the lazy dog. → (the, quick) (the, brown)

The quick brown fox jumps over the lazy dog. → (quick, the) (quick, brown) (quick, fox)

The quick brown fox jumps over the lazy dog. → (brown, the) (brown, quick) (brown, fox) (brown, jumps)

The quick brown fox jumps over the lazy dog. → (fox, quick) (fox, brown) (fox, jumps) (fox, over)

Continues Bag of Words

1. The **CBOW model** architecture tries to predict the current target word (the center word) based on the source context words (surrounding words).
2. The **bag-of-words** model is used to represent an unordered collection of **words** as a vector.
3. One of the most common uses is for simple document classification, an example of this might be the task of classifying an email as spam.

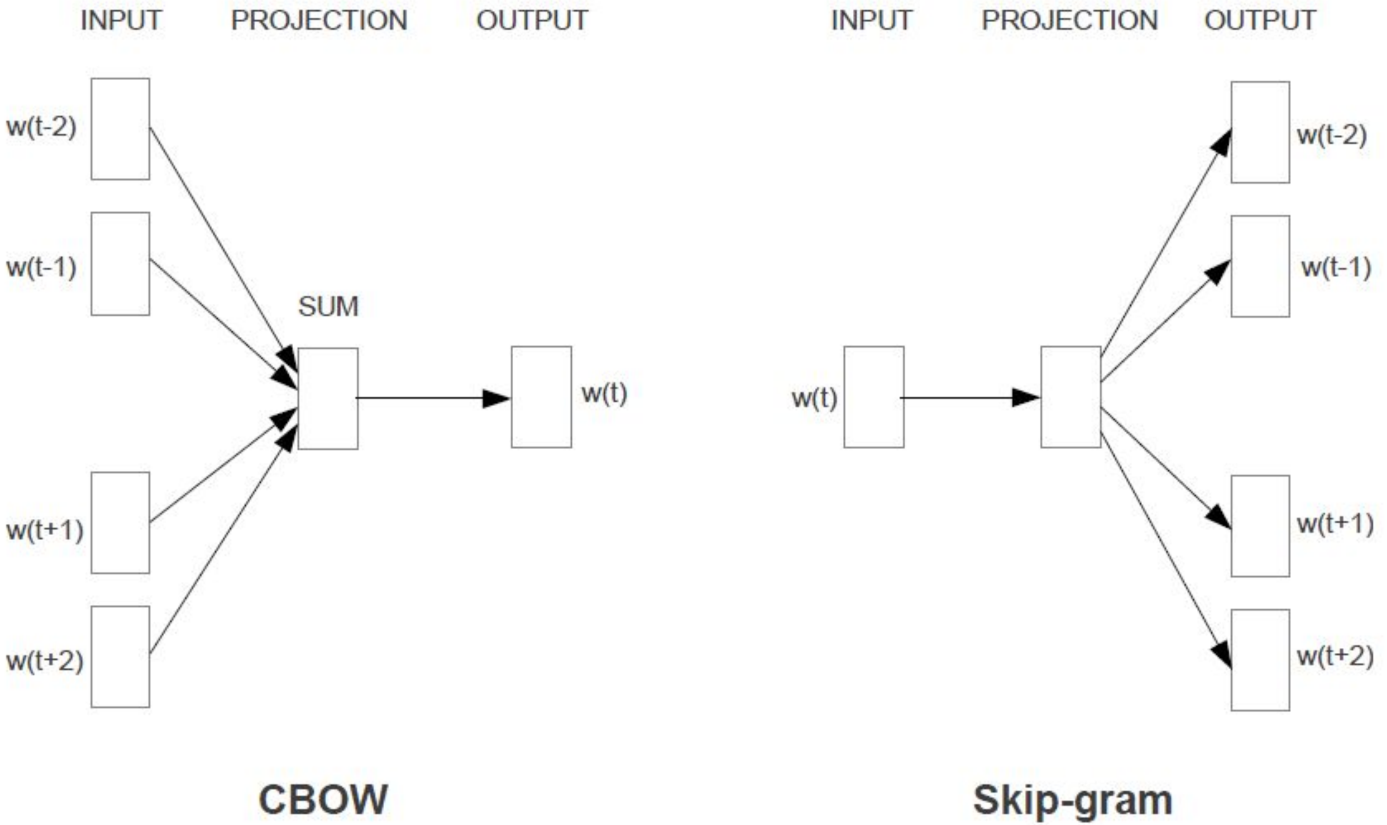


Figure 1: New model architectures. The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word.

GloVe: Global Vectors

The Global Vector (GloVe) model aims to combine the count-based matrix factorization and the context-based skip-gram model together.

Any Questions and Doubts?

We will go for Practical Session :):):)

Limitations in Traditional Language Modelling

The **embeddings are not context-specific** — they are learned based on word concurrency but not sequential context.

So in two sentences,

“*I am eating an apple*” and “*I have an Apple phone*”,

two “apple” words refer to very different things but they would still share the **same word embedding vector.**

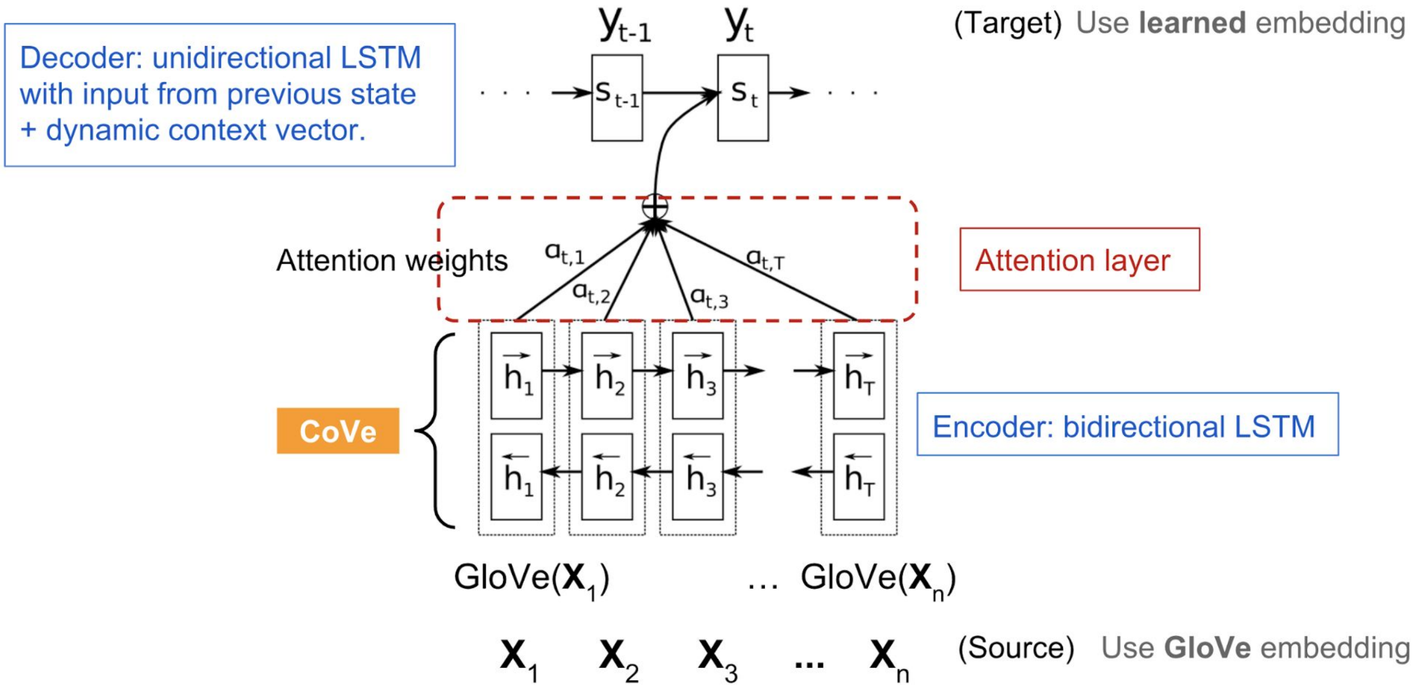
Generalized Language Models

we will discuss how various approaches were proposed to **make embeddings dependent on context**, and to make them easier and cheaper to be **applied to downstream tasks** in general form.

- CoVe
- ELMo
- OpenAI - GPT
- BERT
- OpenAI - GPT -2
- Xlnet

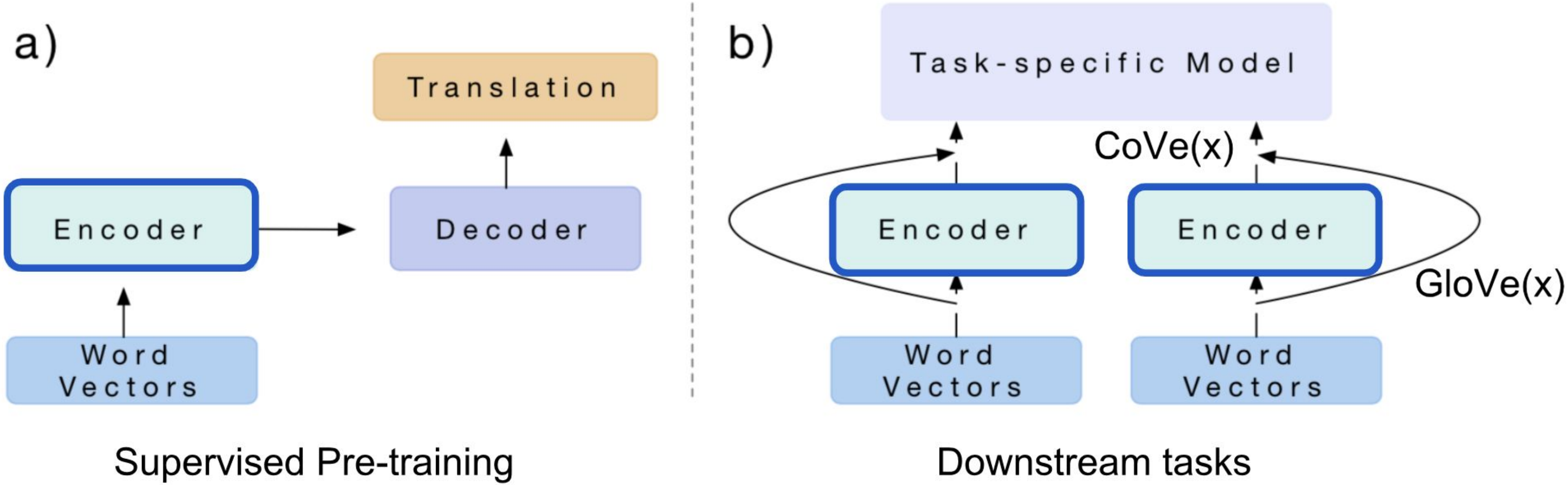
CoVe

CoVe short for **Contextual Word Vectors**, is a type of word embeddings learned by **an encoder in an attentional seq-to-seq machine translation model.** CoVe word representations are functions of the entire input sentence.



Decoder: unidirectional LSTM with input from previous state + dynamic context vector.
y_{t-1}
y_t
s_{t-1}
s_t
(Target) Use **learned** embedding
Attention weights
$a_{t,1}$
$a_{t,2}$
$a_{t,3}$
$a_{t,T}$
Attention layer
CoVe
$\overrightarrow{h}_1$
$\overrightarrow{h}_2$
$\overrightarrow{h}_3$
$\overrightarrow{h}_T$
$\overleftarrow{h}_1$
$\overleftarrow{h}_2$
$\overleftarrow{h}_3$
$\overleftarrow{h}_T$
Encoder: bidirectional LSTM
GloVe($\mathbf{X}_1$) … GloVe($\mathbf{X}_n$)
$\mathbf{X}_1$ $\mathbf{X}_2$ $\mathbf{X}_3$ ... $\mathbf{X}_n$
(Source) Use **GloVe** embedding

Use CoVe in Downstream Tasks



Given a downstream task, we first generate the concatenation of GloVe + CoVe vectors of input words and then feed them into the task-specific models as additional features.

The limitation of CoVe is obvious

- pre-training is bounded by **available datasets** on the supervised translation task;
- the contribution of CoVe to the final performance is constrained by the task-specific model architecture.

we will see that **ELMo overcomes** issue (1) by **unsupervised pre-training**

and **OpenAI GPT & BERT** further overcome both problems by ***unsupervised pre-training + using generative model architecture for different downstream tasks.***

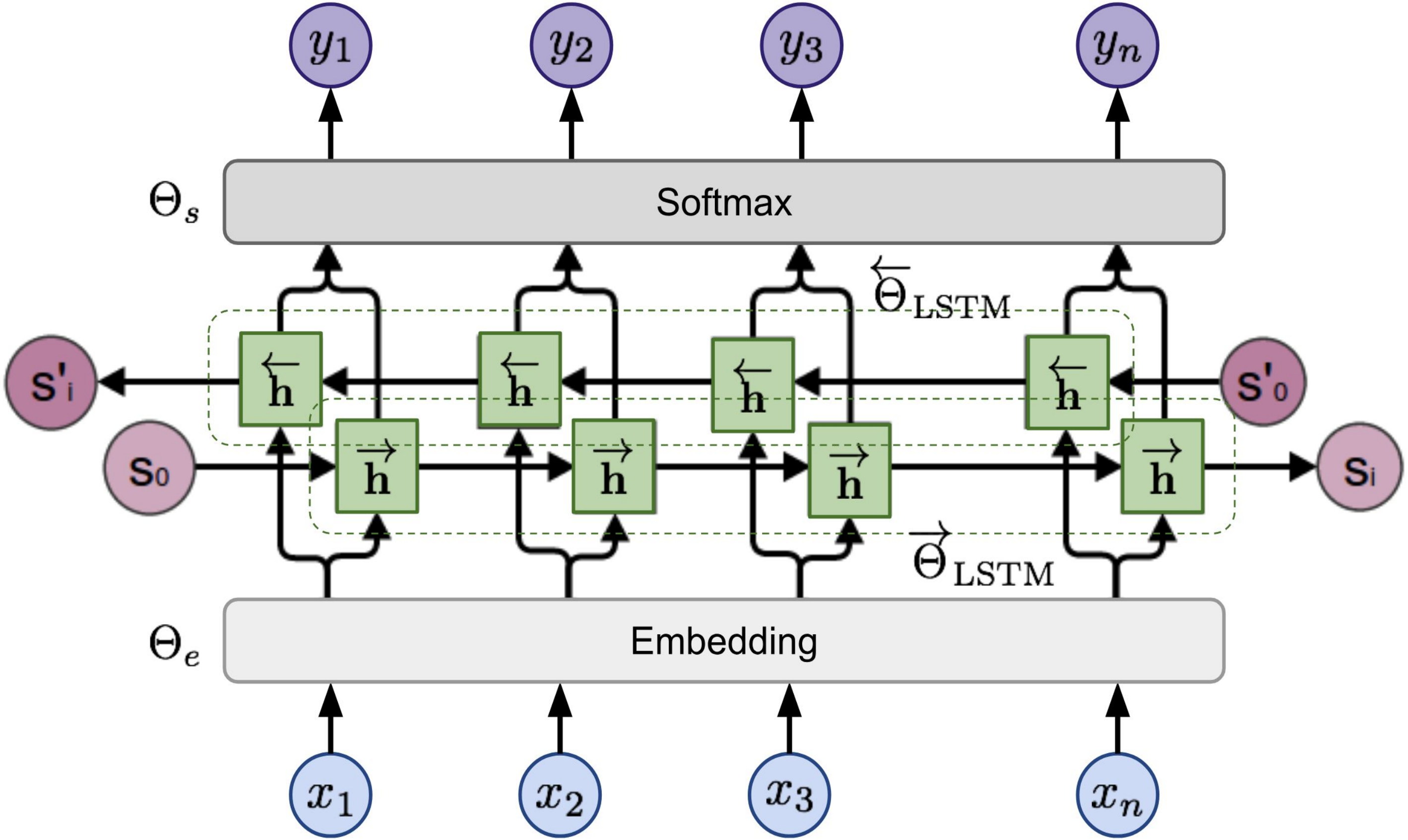
ELMo

ELMo, short for **Embeddings from Language Model** learns contextualized word representation by pre-training a language model in an *unsupervised* way.

Bidirectional Language Model

The bidirectional Language Model (**biLM**) is the foundation for ELMo. While the input is a sequence of n tokens (x1,…,xn), the language model learns to predict the probability of next token given the history.

The biLSTM base model of ELMo


y_1
y_2
y_3
y_n
Θ_s
Softmax
$\overleftarrow{\Theta}_{\text{LSTM}}$
s'_i
$\overleftarrow{\mathbf{h}}$
s'_0
s_0
$\overrightarrow{\mathbf{h}}$
s_i
$\overrightarrow{\Theta}_{\text{LSTM}}$
Θ_e
Embedding
x_1
x_2
x_3
x_n

Use ELMo in Downstream Tasks

ELMo embedding vectors are included in the input or lower levels of task-specific models. Moreover, for some tasks (i.e.,SQuAD, NER), adding them into the output level helps too.

The improvements brought up by ELMo are largest for tasks with a small supervised dataset. With ELMo, we can also **achieve similar performance with much less labeled data.**

Summary: The language model pre-training is unsupervised and theoretically the pre-training can be scaled up as much as possible since the unlabeled text corpora are abundant.

Transformers - Language Models

1. OpenAI - GPT, GPT-2
2. BERT

What is Transformers?

We will discuss here.