Natural Language Processing (NLP) Techniques

Abebaw E.
Adludio, 10 academy

(DL Indaba-X 2021 Ethiopia, AASTU)

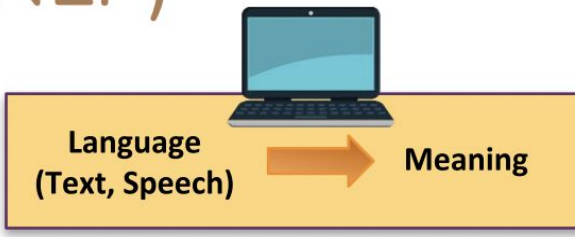
Acknowledgement

- Sujit Pal
- Substain Ruder

Reflection

- What linguistic problem can you see from the statement?

Natural Language Processing (NLP)



NLP is a subfield in AI, where the goal is

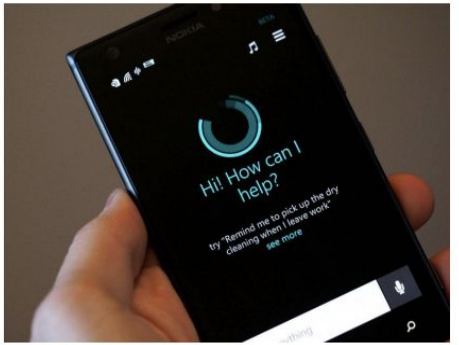
- To bridge the gap between how people **communicate** and what machines **understand** in order to perform useful tasks, e.g. making appointments, buying things, question answering, etc.

NLP (Interpretation)

I want to eat Japanese food.

May I order Yayoi for you?

AI/ML (Generation)



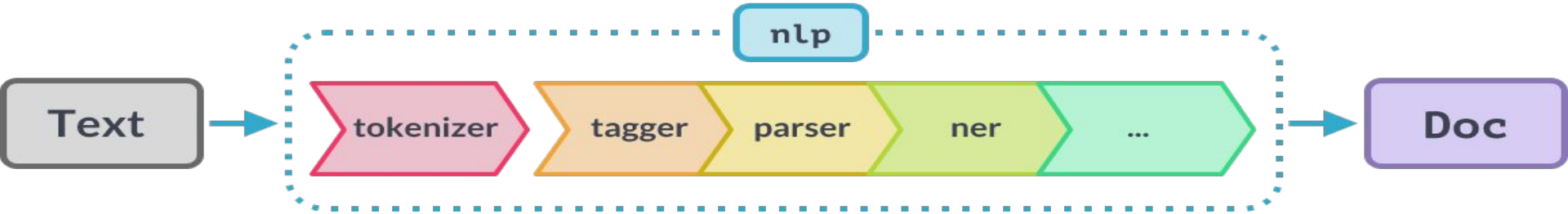
What is NLP?

Automating the analysis, generation, and acquisition of human (“natural”) language

– **Analysis** (or “understanding” or “processing” ...) :input is language, output is some representation that supports useful action

– **Generation:** input is that representation, output is language

– **Acquisition**: obtaining the representation and necessary algorithms, from knowledge and data



Level of understanding in NLP

https://www.tutorialspoint.com/artificial_intelligence/artificial_intelligence_natural_language_processing.htm

Lexical Analysis:

Text→ Paragraphs, Sentences, and Words

Syntactic Analysis (Parsing):

Grammar/Relationship between words

Semantic Analysis:

Exact meaning of the sentence

Discourse Integration:

Meaning of the sentence **based on the previous sentence (pronouns)**

Pragmatic Analysis:

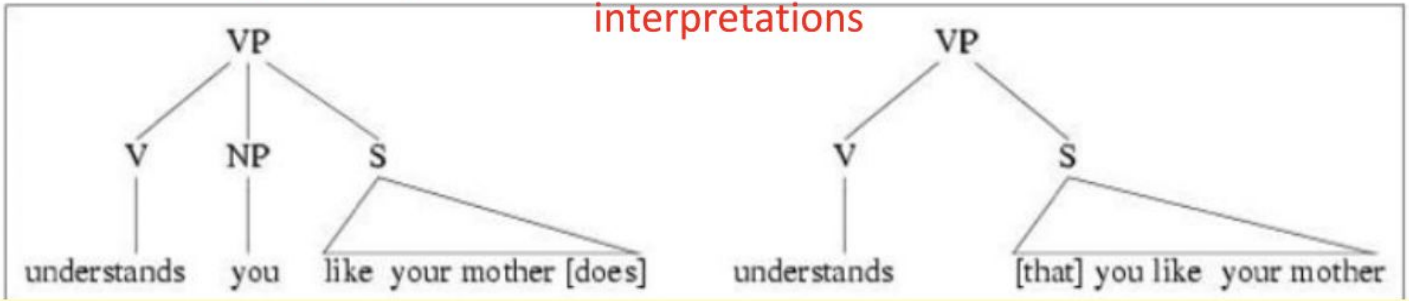
Actual Meaning based on **the context** and real-world knowledge

Why is NLP challenging?

- Human languages are ambiguous (unlike programming and other formal languages), so some parts can be ignored.
- Human languages are interpretation depends on real world, common sense, and contextual knowledge (pragmatic analysis)

At last, a computer understands you like your mother”

Ambiguity at syntactic level: Different structures lead to different interpretations



The Pope’s baby steps on gays. [Ref: Prof. Christopher Manning, CS224N/Ling284, 2017]

Data in NLP

- Format
- Source

Major issues

- Availability: how to solve the problem?
  - Is data available for NLP?
  - Why don't we focus on generating it? – **not possible to work on designing bot application to collect labeled data for some tasks?**
- Quality: how to solve the issue?
  - Metrics?
  - **Should be tested with noise input**
- Balanced data: how to solve the problem?
  - How to know the size?, Representation?, What metrics are used?

Data sheet for dataset

Common preprocessing in NLP

Noise Removal: Remove whitespaces, Remove HTML tags, Convert accented characters, Remove special character, Remove stop words (not always)

Normalization: Lowercasing characters, Expand Contractions, Numeric words to numbers

Stemming / Lemmatization

Vocabulary: get unique tokens from corpus with their frequencies if needed

Vectorization:Bag of word or Word Embedding techniques

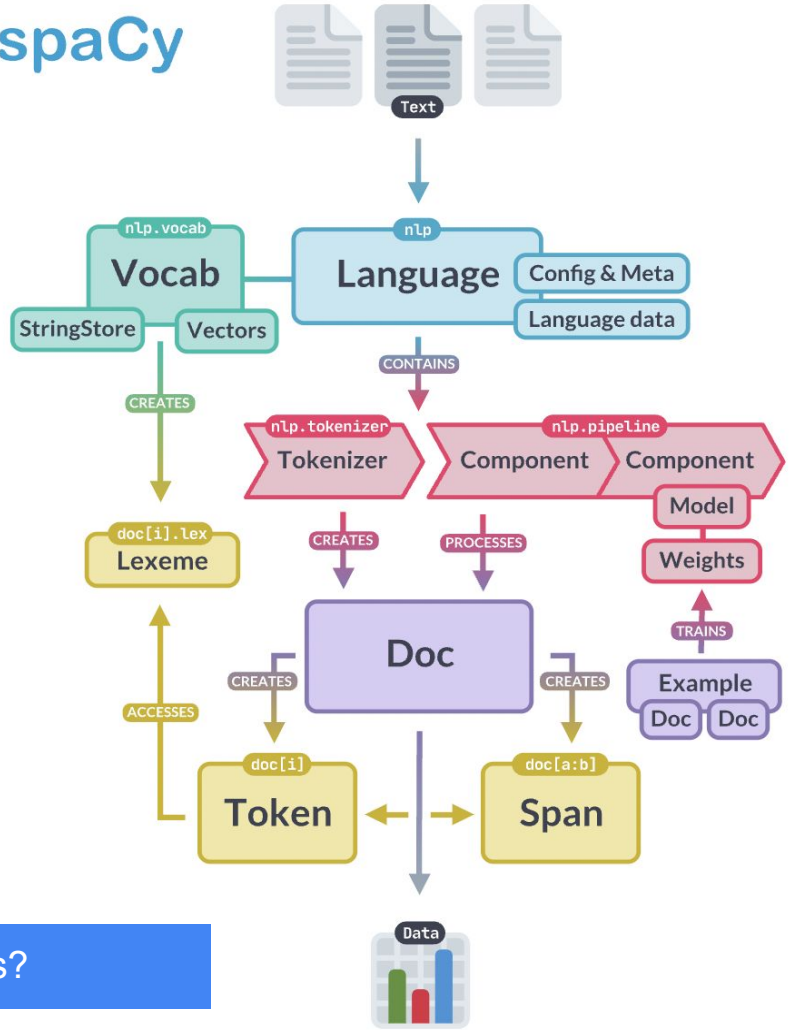
Common Preprocessing tools

- State-of-the-art at NLP
  - spaCy: (Industry solution, cheatsheet)
  - Stanza: Neural pipeline (stanford solution)
  - Huggingface: known to be used as hub for SOTA pre-trained models
- NLTK

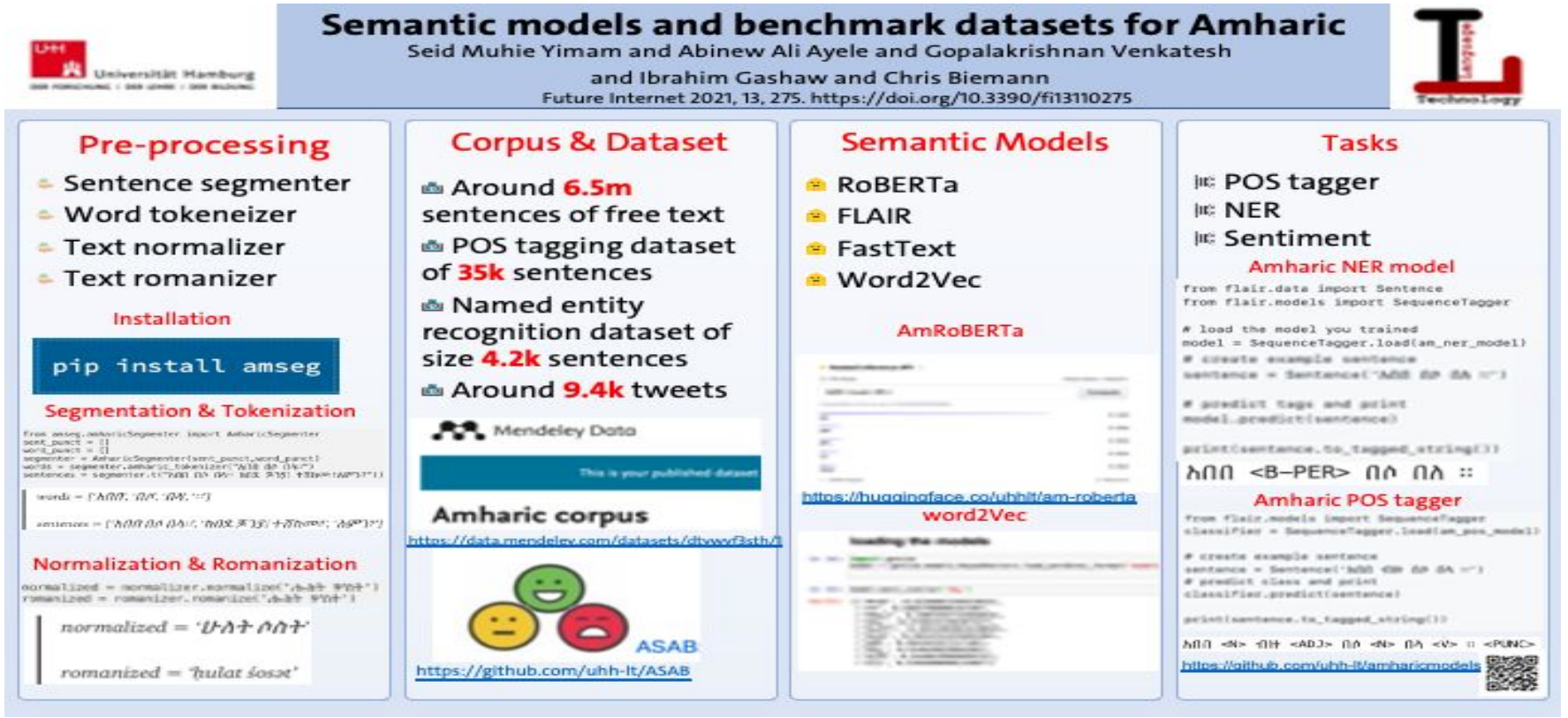
Ethiopic Language tools: Great startup

- AMSEG: Amharic word segmentation model

Not possible to work on such pipeline for local languages?



https://github.com/uhh-lt/amharicmodels



NLP Pipeline

- Statistical Modeling
- Word/character modeling
- Sentence/Document Representation

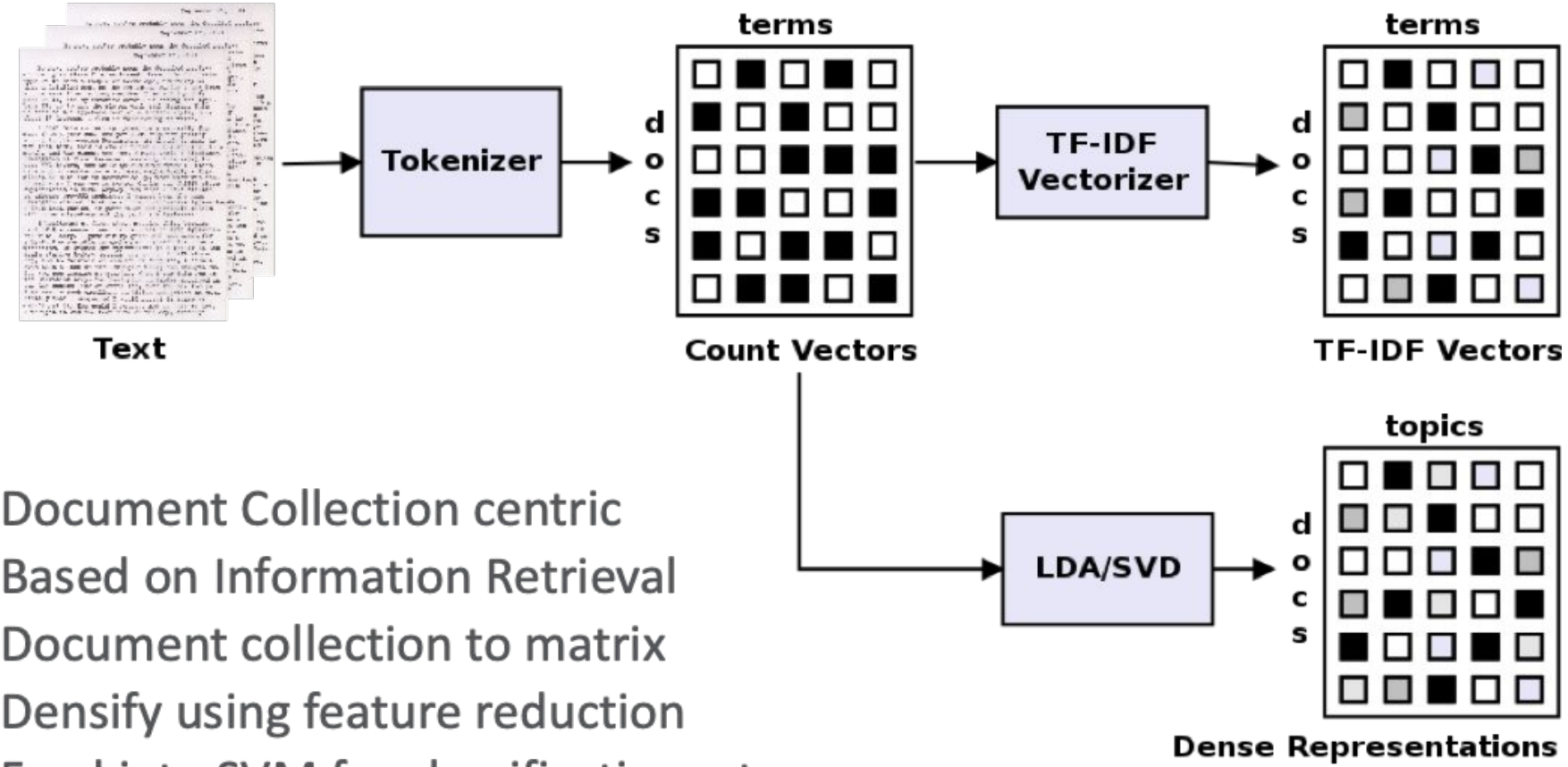
We only focus on text processing for this tutorial

Journey of NLP models

1. **Linear Algebra** – this is a mathematical component required to build a strong base in visualizing the conversion of text to machine understandable format. It helps to correlate how each feature contributes to dimensions used in text representation.
2. **Token Representation (Frequency Based)** – a way to represent a token that can be interpreted by the machine. This transformation could include various dimensions of the text like syntactic, semantic, linguistic, morphological, etc. e.g., Bag of words or TF-IDF
3. **Neural Networks** – these are the connection between neurons present in different layers that help a machine learn some rules based on the input and output data. They are used as building blocks in many state-of-the-art modeling techniques in NLP, hence it is vital to understand the working of neural networks and their different variants.
4. **Transfer Learning** – it is a technique that utilizes the learnings (can be informed of weights or embeddings) from other systems or models. It has proved to be valuable in solving many problems, since you carry forward the learnings and helps to reach an optimum solution in optimal time.

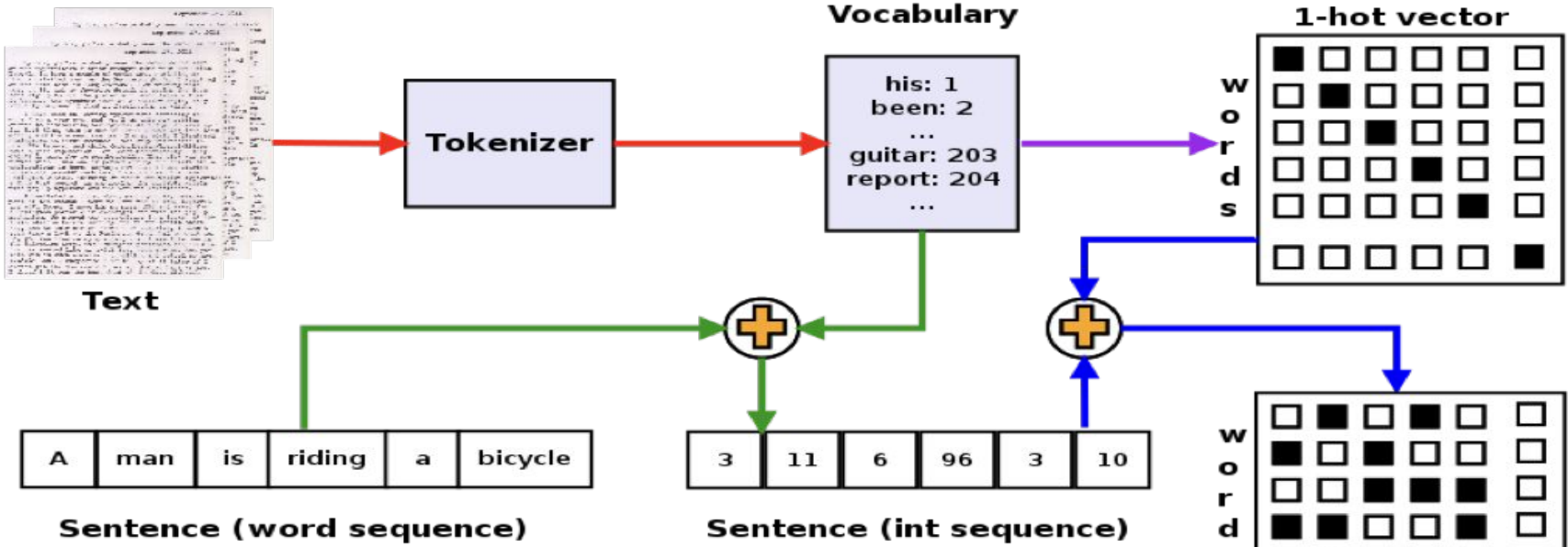
Fig. Evolution of NLP Models

NLP pipeline Frequency based Representation



- Document Collection centric
- Based on Information Retrieval
- Document collection to matrix
- Densify using feature reduction
- Feed into SVM for classification, etc.

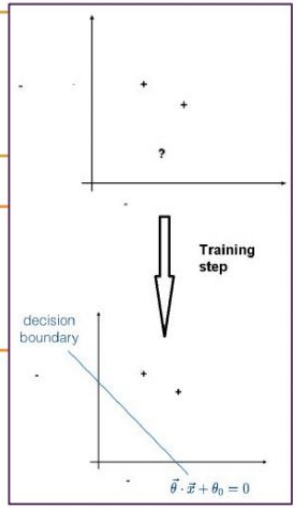
NLP pipeline Sparse Matrix Representation



- Idea borrowed from Machine Learning (ML)
- Represent categorical variables (words) as 1-hot vectors
- Represent sentences as matrix of 1-hot word vectors
- No distributional semantics at word level.

Limitation of traditional statistical approach

1. • Data collection
   • Obtain the labels {-1, +1}
2. • Feature Extraction
   • Most common features in NLP is word n-grams → encoded as one-hot vector
3. • Training
4. • Testing



- Sparsity:
  - feature vectors are typically high-dimensional and sparse (i.e. most elements are 0).
- Feature engineering:
  - Need experts to manually

Map **discrete**, one-hot vectors into low-dimensional **continuous** representations.
***** Self learned features → Deep Learning *****

pear

$[1\ 0\ 0\ 0\ \ldots\ 0]$

↓

$[0.4\ 0.1\ 0.1]$

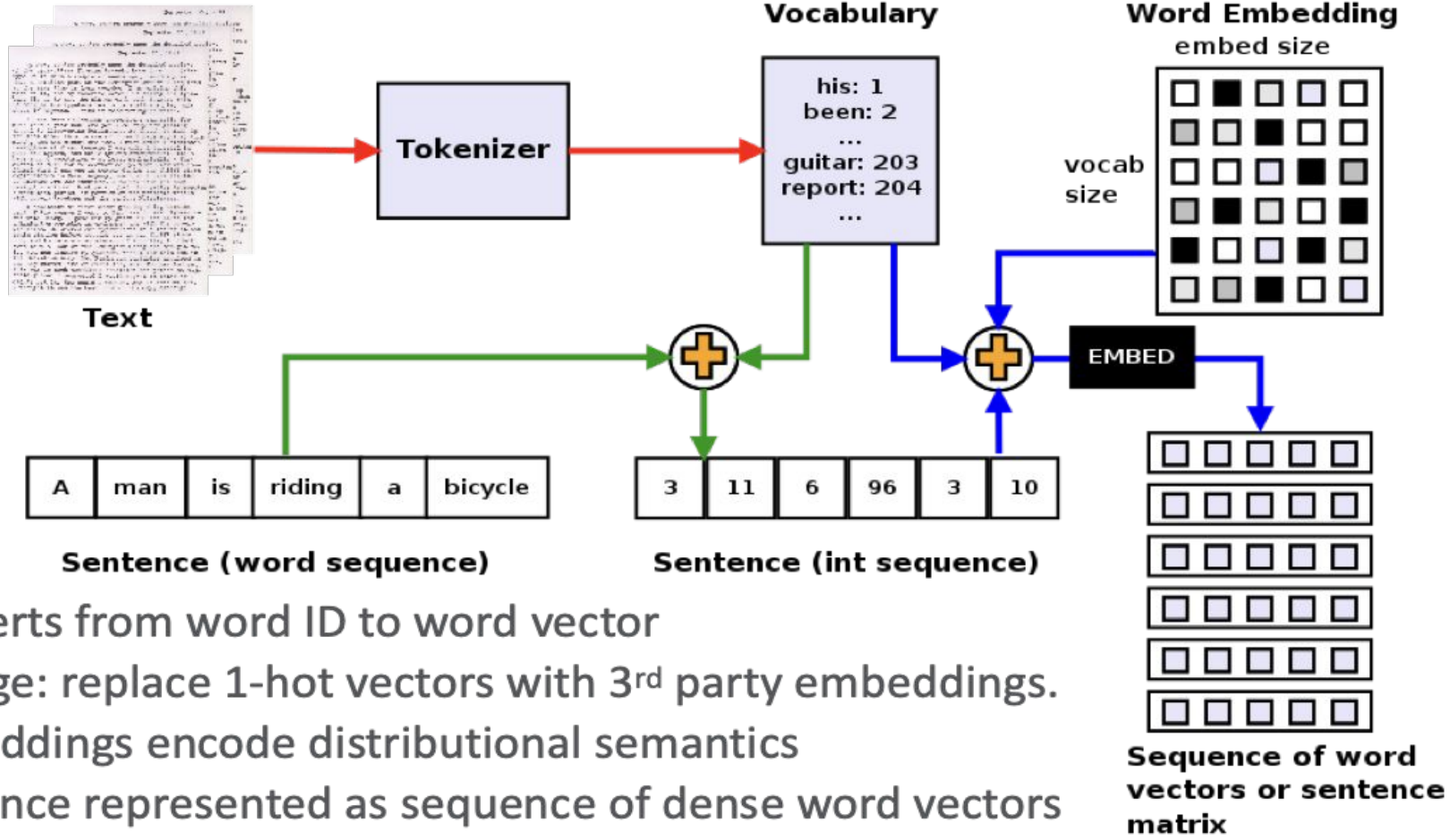
apple

$[0\ 0\ 1\ 0\ \ldots\ 0]$

↓

$[0.6\ 0.2\ 0.3]$

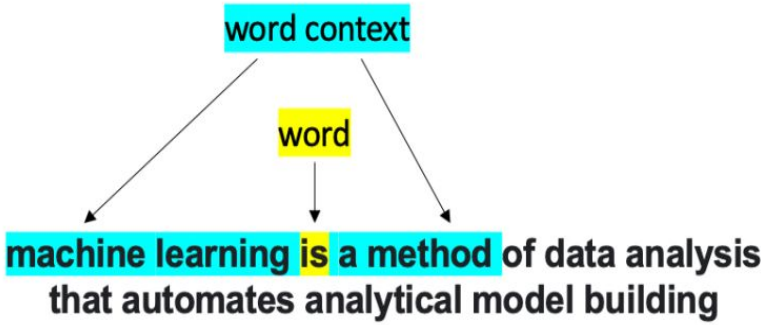
NLP pipeline with word context representation models



- Converts from word ID to word vector
- Change: replace 1-hot vectors with 3rd party embeddings.
- Embeddings encode distributional semantics
- Sentence represented as sequence of dense word vectors

From words to words-in-context

- ***Semantics*** is concerned with the meanings of texts
- Can be achieved:
  - Propositional ***or formal semantics***: A block of text is to converted into a formula in a logical language, e.g. predicate calculus.
  - **vector Representation**: Texts are ***embedded*** into a high-dimensional space. (our focus)

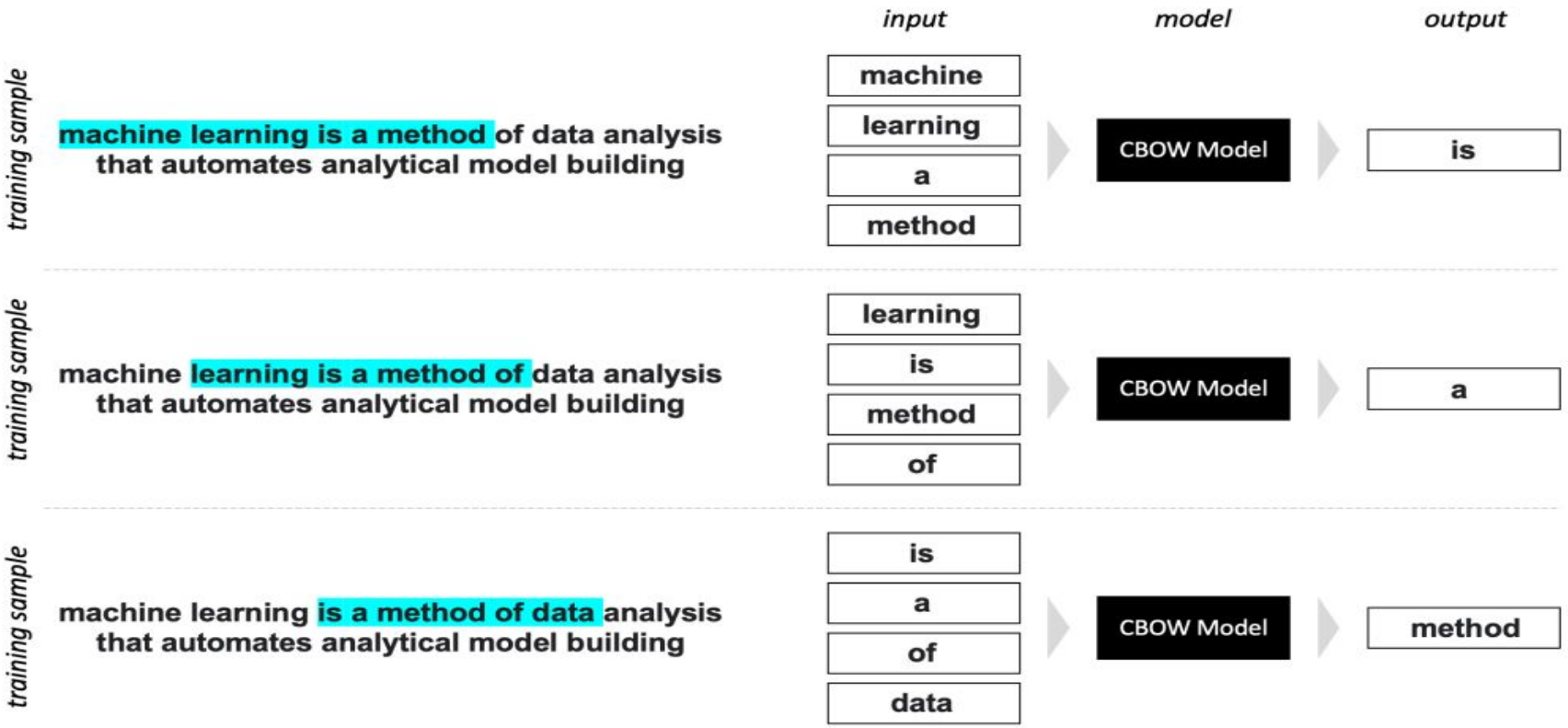


Classic Word Context Embedding Models

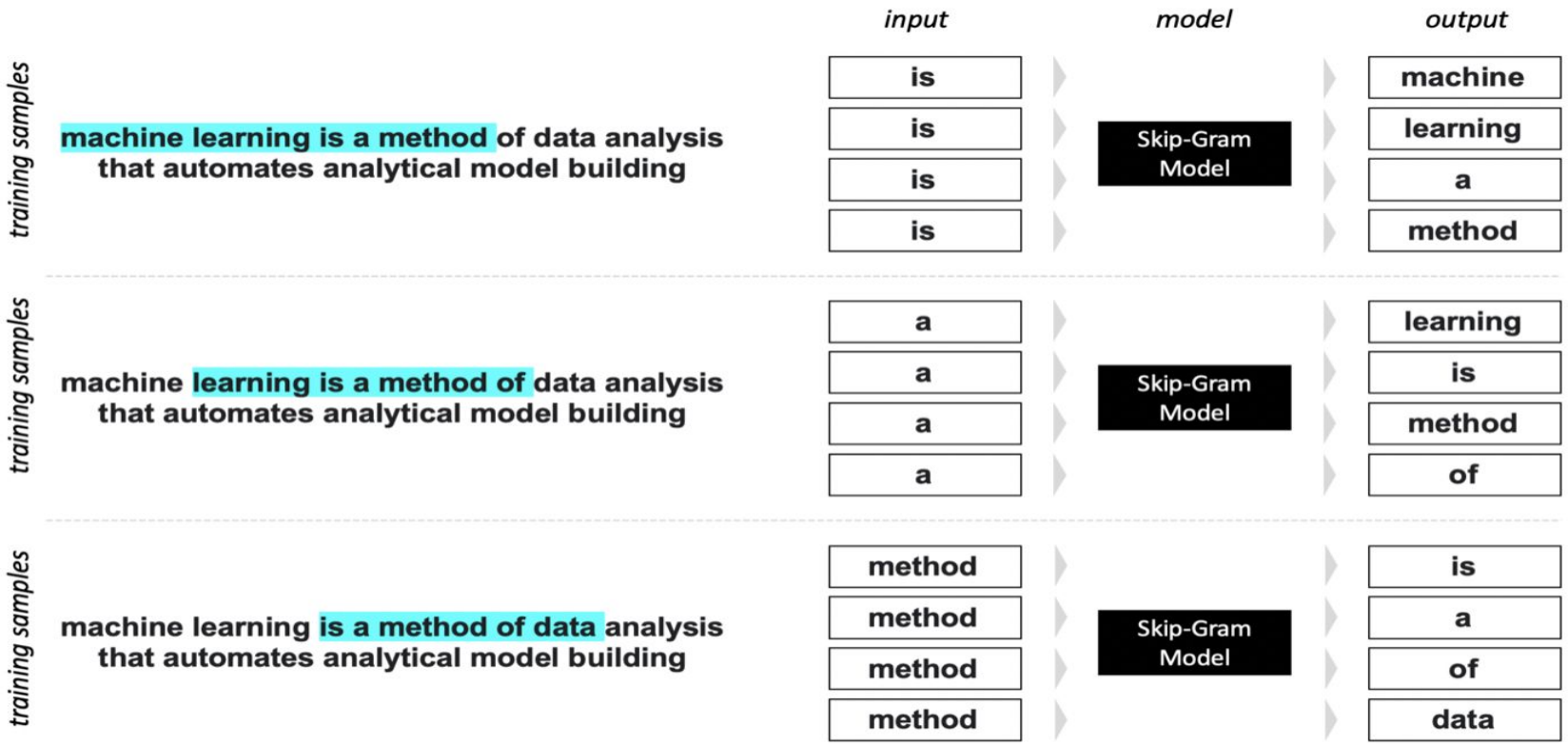
Pre-trained over very large corpora and shown to capture latent syntactic and semantic features.

- **Word2vec (Google)**: has two model architectures to produce word vectors (Both methods are trained based on a neural prediction-based model)
  - **continuous bag-of-words (CBOW)**: trains a model that aims to predict a word given its context
  - **skip-gram** : trains a model that aims to predict the context given the centre word in it.
- **FastText(Facebook)**: embeds words by treating each word as being composed of character n-grams instead of a word whole.
- **GloVe(Stanford)**: learns efficient word representations by performing training on aggregated global word-word co-occurrence statistics from a corpus.

CBOW Word2Vec Model



Skip-Gram Word2Vec model



Generate Vocabulary

Preprocess Text

Take train dataset. Lowercase the all text and split it into tokens.

Example:

BEFORE
Machine learning is a method of data analysis that automates analytical model building.

AFTER
[`machine`, `learning`, `is`, `a`, `method`, `of`, `data`, `analysis`, `that`, `automates`, `analytical`, `model`, `building`, `.`]

Calculate Word Frequencies

For every word in the dataset, calculate ts number of occurrences.

Example:

Word	N occurrences
a	4052
apple	553
brown	3
dog	85
is	642
learning	45
mammal	39
method	89
they	1263
university	652
zoo	4
...	...

Choose Words for Vocabulary

From most frequent words create a Vocabulary.

Example:

Word	N occurrences
a	4052
apple	9
brown	3
dog	85
is	642
learning	45
mammal	39
method	89
they	1263
university	12
zoo	4
...	...

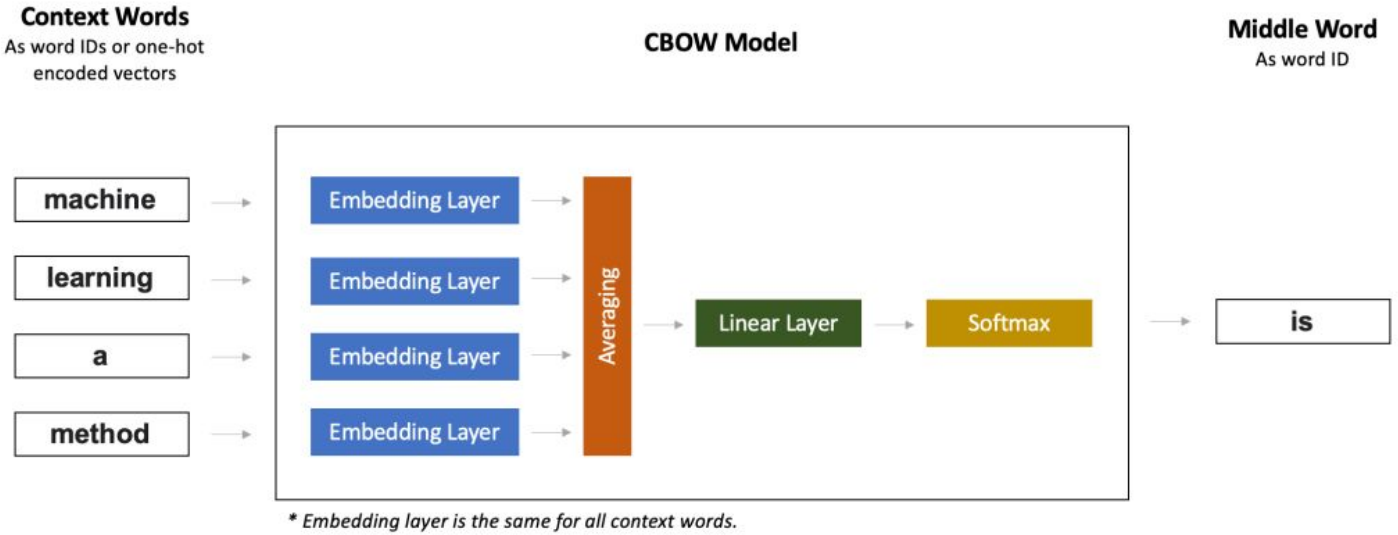
Encode Vocabulary Words

Assign ID to each word. Assign ID=0 to out-of-vocabulary words.

Example:

Word	ID
<unk>	0
a	1
dog	2
is	3
learning	4
mammal	5
method	6
they	7

Word2vec walkthrough



Embedding layer: a simple lookup table with learnable weights, or as a linear layer without bias and activation.

Linear (Dense) layer: creates multi-class classification task, where the number of classes is equal to the number of words in the vocabulary. This layer uses Softmax activation

Difference:

- CBOW model takes n-words, each goes through the same Embedding layer, and then word embedding vectors are averaged before going into the Linear layer.
- Skip-Gram model takes a single word instead.

Laboratory use cases

- Create preprocessing pipeline for Amharic
- Word and subword representation model for Amharic
- Creating model, visualizing and uploading models

Best Practices for morphologically rich languages

- Sub-word learning (our amharic work).
- Word2vec with character level models
- Concatinate word vectors with word classes (tags or any semantic info) vectors or add as auxulary input at attention layer
- Depth preprocessing
  - Try with or without lowlevel word precessings (stemming, lematizer, ..)
  - Sometimes character level models outperform by detecting misspeled words

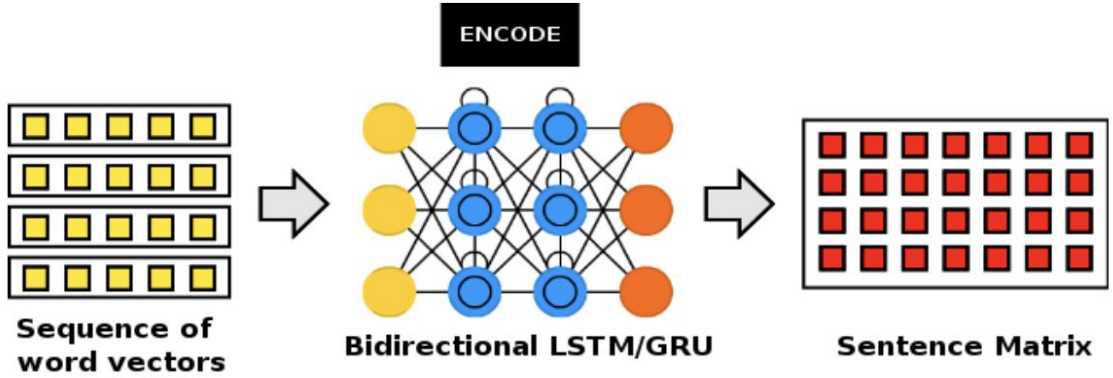
Part 2: Advanced NLP Techniques

- Seq2seq
- Attention mechanisms
- Transformers
- Recent directions

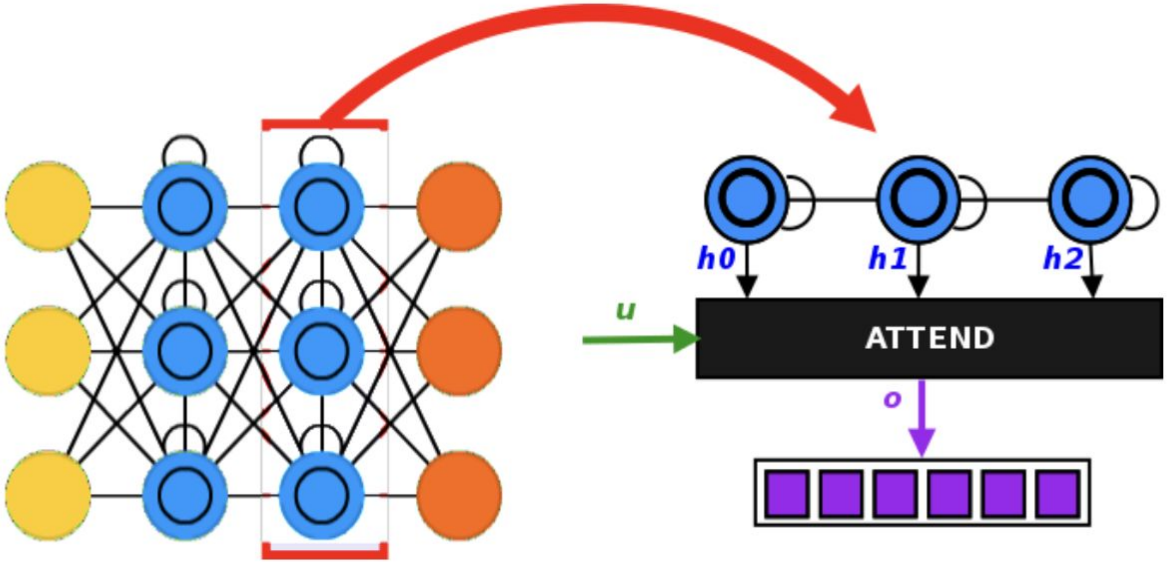
Sequence-to-sequence modeling in NLP

- **Why RNN fits the need of sentence and document encoding?**

Encoding



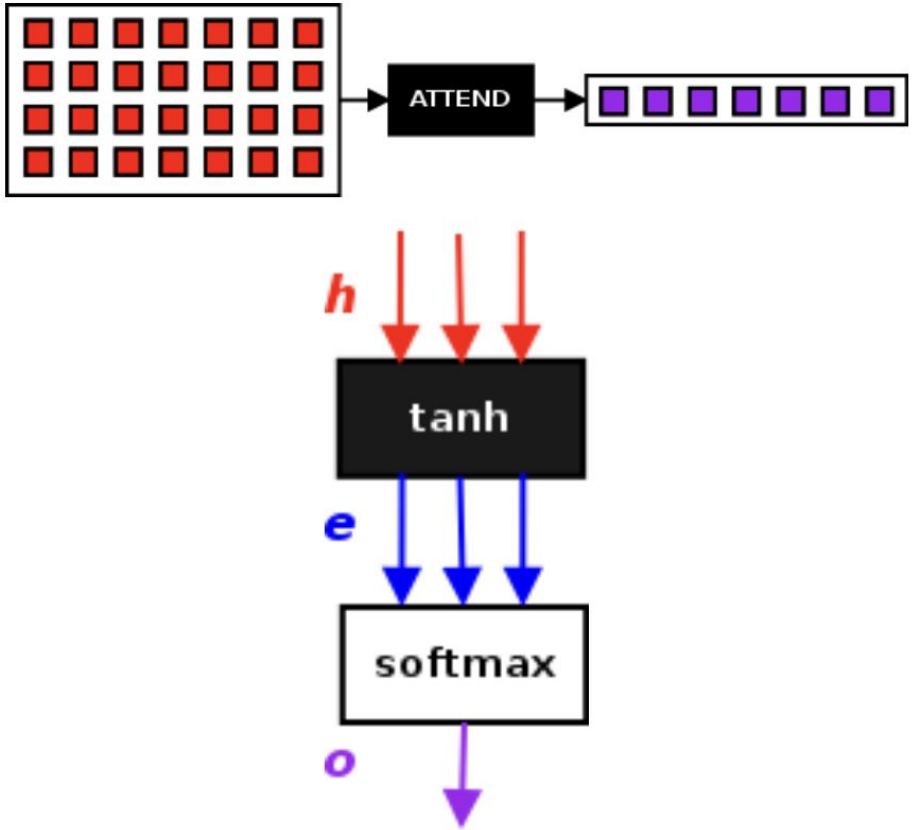
- Converts sequence of vectors (word vectors) to a matrix (sentence matrix).
- Bag of words – concatenate word vectors together.
- Each row of sentence matrix encodes the meaning of each word in the **context of the sentence**.
- Generally use LSTM (Long Short Term Memory) or GRU (Gated Recurrent Unit)
- Bidirectional processes words left to right and right to left and concatenates.



- Reduces matrix (sentence matrix) to a vector (sentence vector)
- Non-attention mechanism –Sum or Average/Max Pooling
- Attention tells what to keep during reduction to **minimize information loss**.
- Different kinds – matrix, matrix + context (learned), matrix + vector (provided), matrix + matrix.

ATTENTION: MATRIX

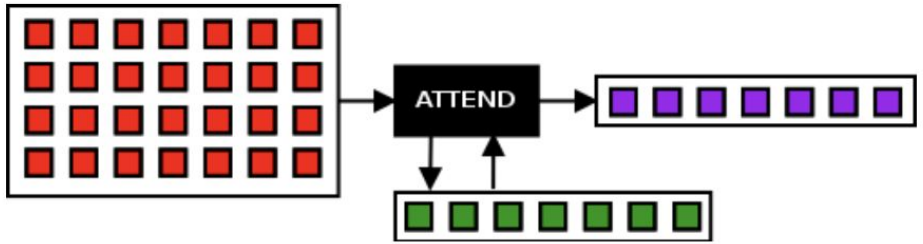
$$e_t = tanh(Wh_t + b)$$
$$\alpha_t = softmax(e_t)$$
$$o = \sum_t \alpha_t h_t$$



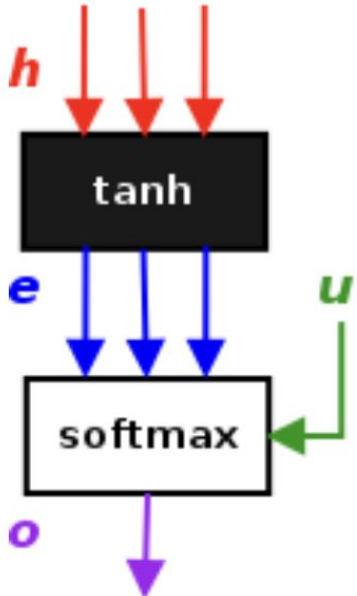
- Proposed by Raffel, et al
- Intuition: select most important element from each timestep
- Learnable weights W and b depending on target.

ATTENTION: MATRIX + VECTOR (LEARNED)

$$e_t = tanh(Wh_t + b)$$
$$\alpha_t = softmax(ue_t)$$
$$o = \sum_t \alpha_t h_t$$



- Proposed by Lin, et al
- Intuition: select most important element from each timestep and weight with another learned vector u.



ATTENTION: MATRIX + MATRIX

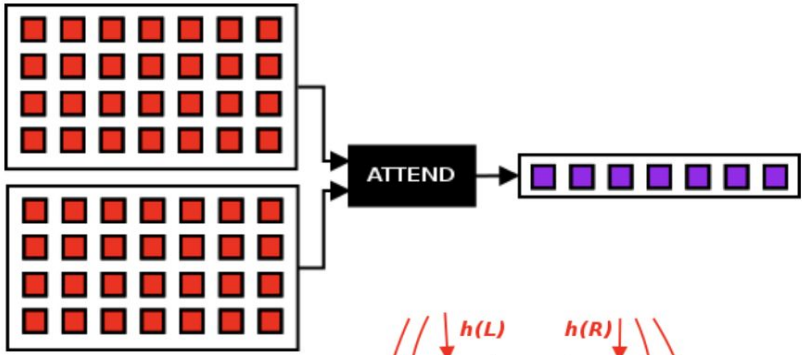
$$e_t^{(L)} = tanh(W_1 h_t^{(L)} + b_1) \qquad e_t^{(R)} = tanh(W_2 h_t^{(R)} + b_2)$$

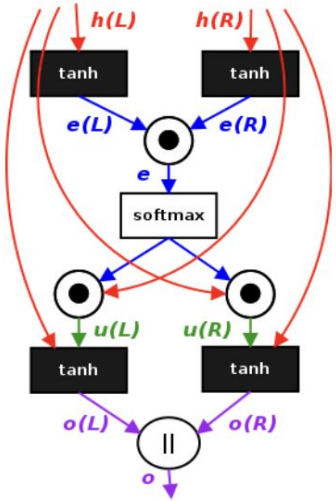
$$e_t = e_t^{(L)}.e_t^{(R)}$$

$$\alpha^{(L)} = \sum_t softmax(e_t).h_t^{(R)} \qquad \alpha^{(R)} = \sum_t softmax(e_t).h_1^{(L)}$$

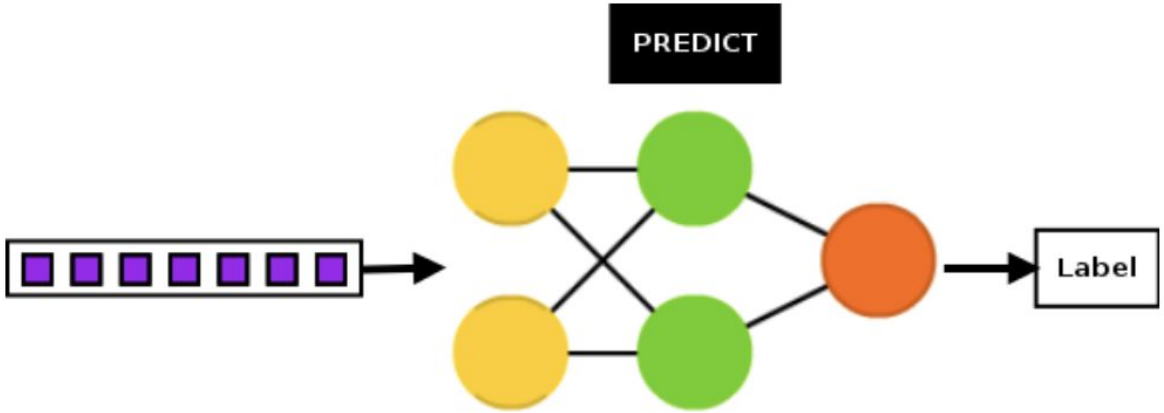
$$o^{(L)} = \sum_t tanh(U_1 \alpha^{(L)} + V_1 h_t^{(L)}) \qquad o^{(R)} = \sum_t tanh(U_2 \alpha^{(R)} + V_2 h_t^{(R)})$$

$$o = concat(o^{(L)}, o^{(R)})$$





- Proposed by Parikh, et al
- Intuition: build alignment (similarity) matrix by multiplying learned vectors from each matrix, compute context vectors from the alignment matrix, and mix with original signal.



- Convert reduced vector to a label.
- Generally uses shallow fully connected networks such as the one shown.
- Can also be modified to have a regression head (return the probabilities from the softmax activation.

Transfer Learning

Transfer learning refers to a machine learning (ML) technique in which a model is trained for the main task and then repurposed for another, similar task.

So, instead of developing and training a new model from scratch, you can just revamp an existing model.

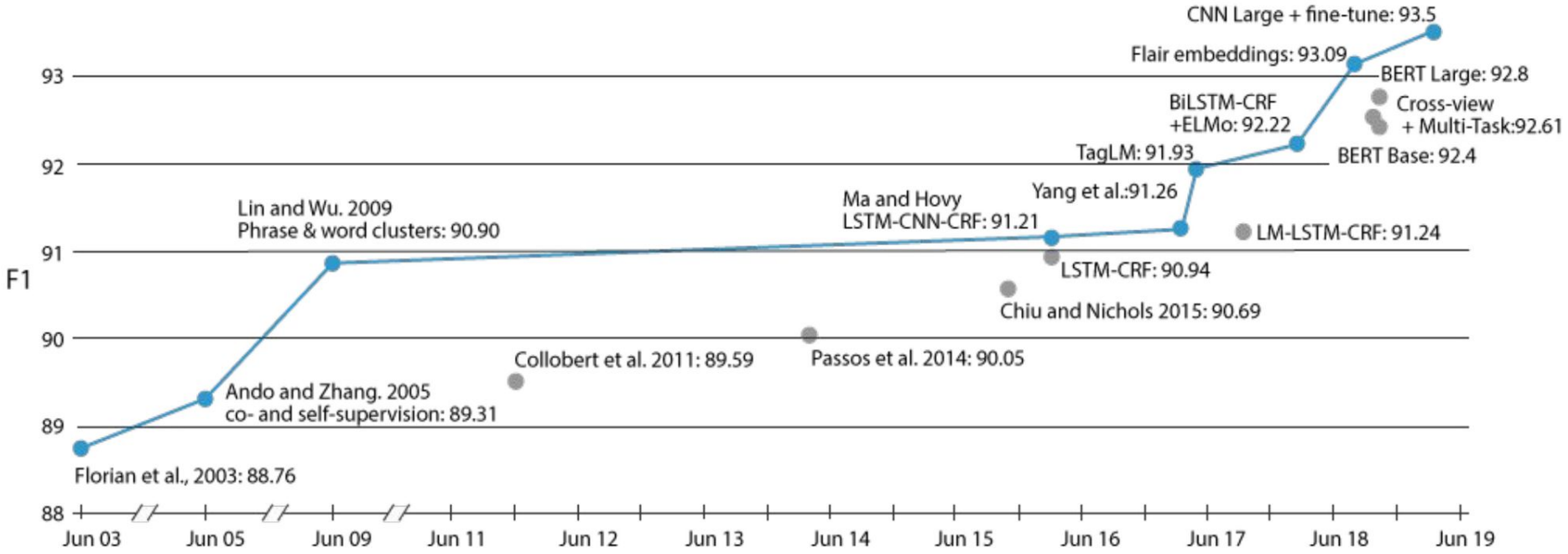
Why Transfer learning in NLP?

- Many NLP tasks share common knowledge about language (e.g. linguistic representations, structural similarities)
- Tasks can inform each other—e.g. syntax and semantics
- Annotated data is rare, make use of as much supervision as available.

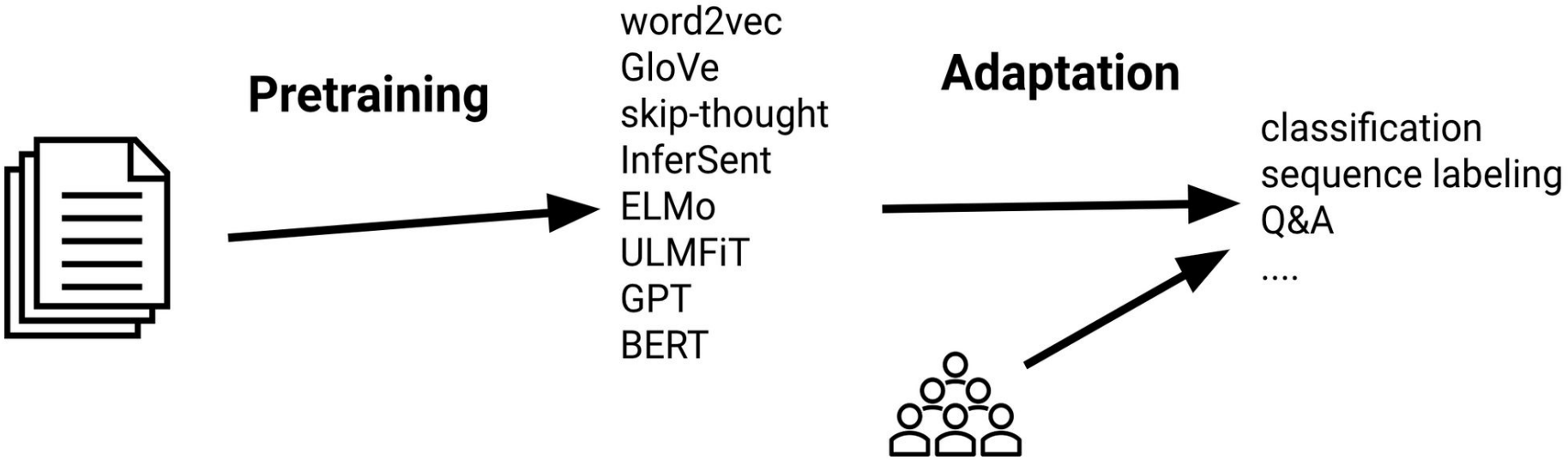
- Empirically, transfer learning has resulted in SOTA for many supervised NLP tasks (e.g. classification, information extraction, Q&A, etc).

Why Transfer learning in NLP? (Emperically)

Performance on Named Entity Recognition (NER) on CoNLL-2003 (English) over time

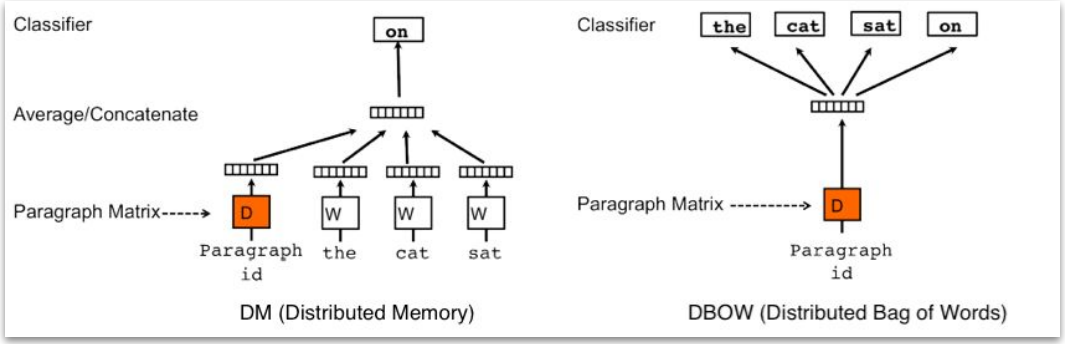


Sequential transfer learning



Paragraph vector

Unsupervised paragraph embeddings (Le & Mikolov, 2014)

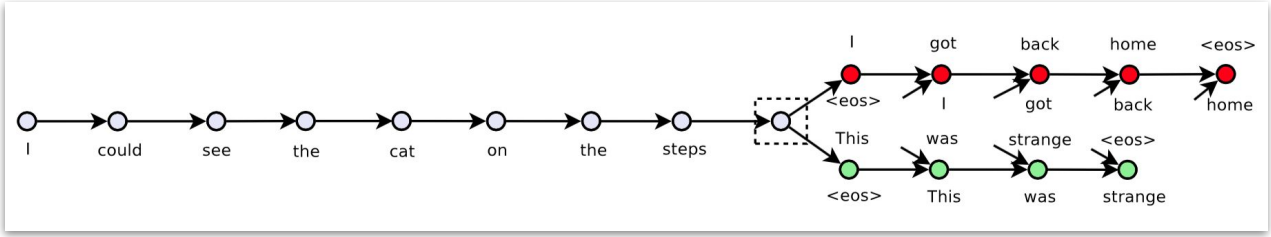


SOTA classification (IMDB, SST)

Model	Error rate
BoW (bnc) (Maas et al., 2011)	12.20 %
BoW (bΔt'c) (Maas et al., 2011)	11.77%
LDA (Maas et al., 2011)	32.58%
Full+BoW (Maas et al., 2011)	11.67%
Full+Unlabeled+BoW (Maas et al., 2011)	11.11%
WRRBM (Dahl et al., 2012)	12.58%
WRRBM + BoW (bnc) (Dahl et al., 2012)	10.77%
MNB-uni (Wang & Manning, 2012)	16.45%
MNB-bi (Wang & Manning, 2012)	13.41%
SVM-uni (Wang & Manning, 2012)	13.05%
SVM-bi (Wang & Manning, 2012)	10.84%
NBSVM-uni (Wang & Manning, 2012)	11.71%
NBSVM-bi (Wang & Manning, 2012)	8.78%
Paragraph Vector	**7.42%**

Skip-Thought Vectors

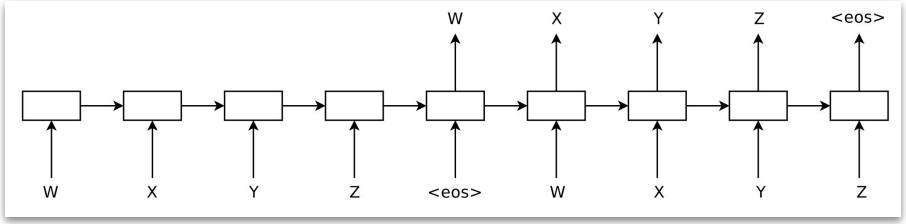
Predict previous / next sentence with seq2seq model (Kiros et al., 2015)



Method	MR	CR	SUBJ	MPQA	TREC
NB-SVM [41]	79.4	81.8	93.2	86.3	
MNB [41]	79.0	80.0	93.6	86.3	
cBoW [6]	77.2	79.9	91.3	86.4	87.3
GrConv [6]	76.3	81.3	89.5	84.5	88.4
RNN [6]	77.2	82.3	93.7	90.1	90.2
BRNN [6]	82.3	82.6	94.2	90.3	91.0
CNN [4]	81.5	85.0	93.4	89.6	**93.6**
AdaSent [6]	**83.1**	**86.3**	**95.5**	**93.3**	92.4
Paragraph-vector [7]	74.8	78.1	90.5	74.2	91.8
uni-skip	75.5	79.3	92.1	86.9	91.4
bi-skip	73.9	77.9	92.5	83.3	89.4
combine-skip	76.5	80.1	93.6	87.1	92.2
combine-skip + NB	80.4	81.3	93.6	87.5	

Hidden state of encoder transfers to sentence tasks (classification, semantic similarity)

Dai & Le (2015): Pretrain a sequence autoencoder (SA) and generative LM



SOTA classification (IMDB)

Model	Test error rate
LSTM with tuning and dropout	13.50%
LSTM initialized with word2vec embeddings	10.00%
LM-LSTM (see Section 2)	7.64%
SA-LSTM (see Figure 1)	7.24%
SA-LSTM with linear gain (see Section 3)	9.17%
SA-LSTM with joint training (see Section 3)	14.70%
Full+Unlabeled+BoW [21]	11.11%
WRRBM + BoW (bnc) [21]	10.77%
NBSVM-bi (Naïve Bayes SVM with bigrams) [35]	8.78%
seq2-bow*n*-CNN (ConvNet with dynamic pooling) [11]	7.67%
Paragraph Vectors [18]	7.42%

See also:

- Socher et. al (2011): Semi-supervised recursive auto encoder
- Bowman et al. (2016): Variational autoencoder (VAE)
- Hill et al. (2016): Denoising autoencoder

Autoencoder pretraining

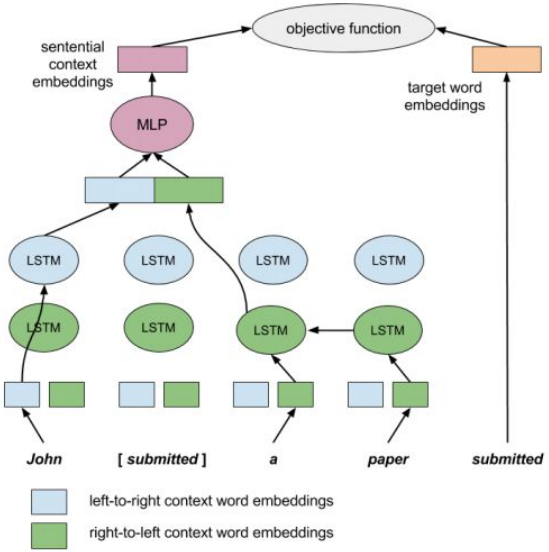
Supervised sentence embeddings

Also possible to train sentence embeddings with supervised objective

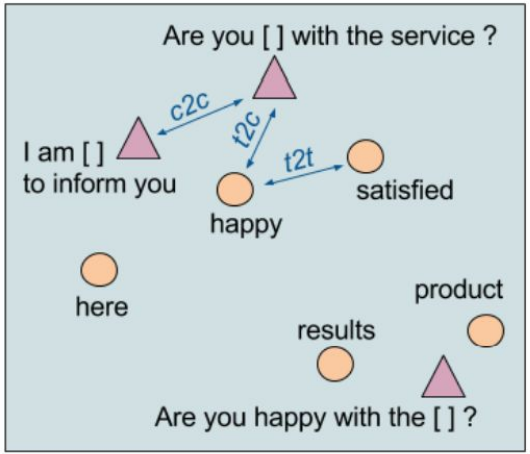
- Paragram-phrase: uses paraphrase database for supervision, best for paraphrase and semantic similarity (Wieting et al. 2016)
- InferSent: bi-LSTM trained on SNLI + MNLI (Conneau et al. 2017)
- GenSen: multitask training (skip-thought, machine translation, NLI, parsing) (Subramanian et al. 2018)

context2vec

Use bidirectional LSTM and cloze prediction objective (a 1 layer masked LM)



Learn representations for both words and contexts (minus word)



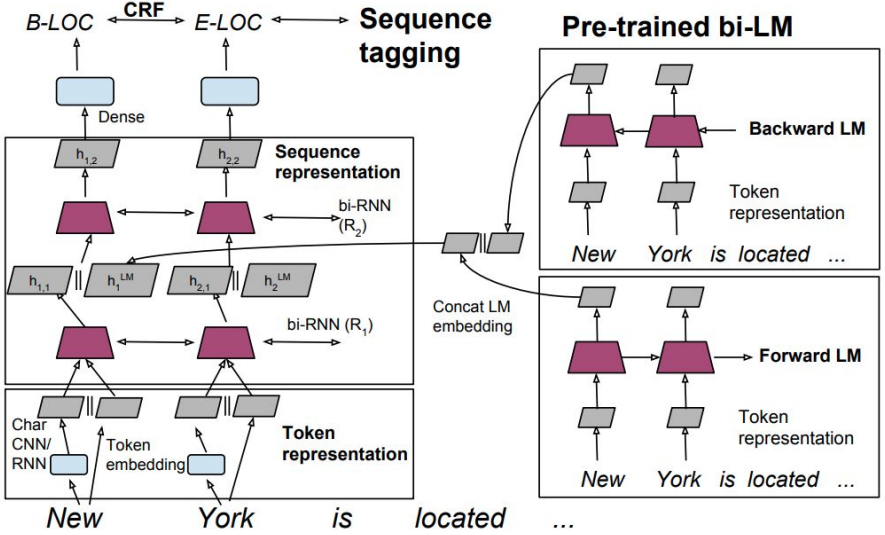
Sentence completion
Lexical substitution
WSD

	c2v iters+	*c2v*	*AWE*	S-1	S-2
MCSS					
test	**64.0**	62.7	48.4	-	-
all	**65.1**	61.3	49.7	58.9	56.2
LST-07					
test	**56.1**	54.8	41.9	55.2	-
all	**56.0**	54.6	42.5	55.1	53.6
LST-14					
test	47.7	47.3	38.1	**50.0**	-
all	47.9	47.5	38.9	**50.2**	48.3
SE-3					
test	72.8	71.2	61.4	**74.1**	73.6

(Melamud et al., CoNLL 2016)

TagLM

Pretrain two LMs (forward and backward) and add to sequence tagger.
SOTA NER and chunking results



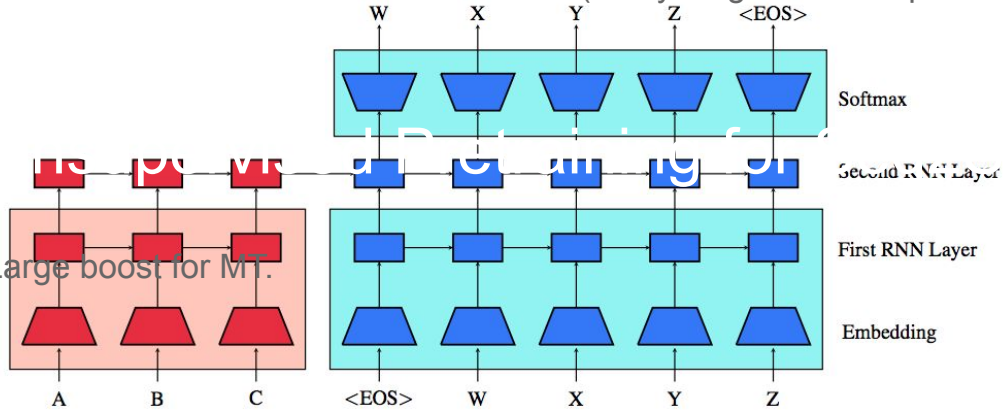
Model	$F_1 \pm$ std
Chiu and Nichols (2016)	90.91 ± 0.20
Lample et al. (2016)	90.94
Ma and Hovy (2016)	91.37
Our baseline without LM	90.87 ± 0.13
TagLM	$\mathbf{91.93 \pm 0.19}$

Table 1: Test set F_1 comparison on CoNLL 2003 NER task, using only CoNLL 2003 data and unlabeled text.

(Peters et al. ACL 2017)

GPT

Pretrain encoder and decoder with LMs (everything shaded is pretrained).

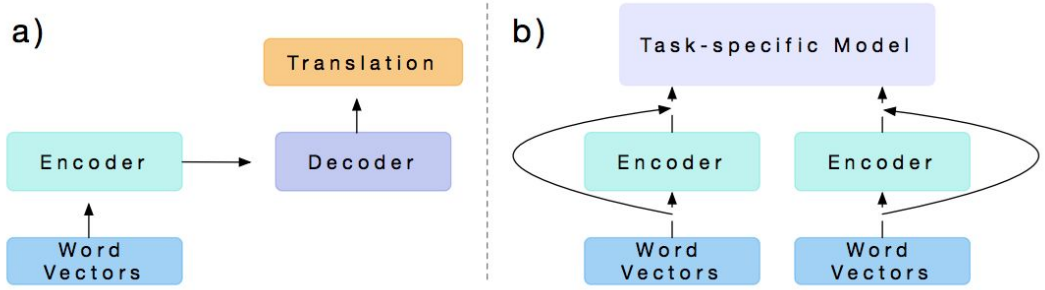


Large boost for MT.

System	ensemble?	BLEU newstest2014	BLEU newstest2015
Phrase Based MT (Williams et al., 2016)	-	21.9	23.7
Supervised NMT (Jean et al., 2015)	single	-	22.4
Edit Distance Transducer NMT (Stahlberg et al., 2016)	single	21.7	24.1
Edit Distance Transducer NMT (Stahlberg et al., 2016)	ensemble 8	22.9	25.7
Backtranslation (Sennrich et al., 2015a)	single	22.7	25.7
Backtranslation (Sennrich et al., 2015a)	ensemble 4	23.8	26.5
Backtranslation (Sennrich et al., 2015a)	ensemble 12	**24.7**	27.6
No pretraining	single	21.3	24.3
Pretrained seq2seq	single	**24.0**	**27.0**
Pretrained seq2seq	ensemble 5	**24.7**	**28.1**

(Ramachandran et al. EMNLP 2017)

CoVe



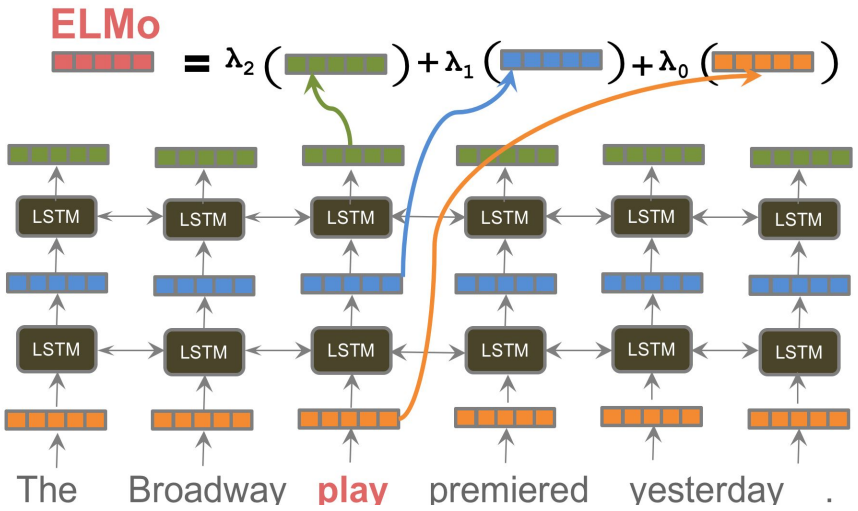
			GloVe+				
Dataset	Random	GloVe	Char	CoVe-S	CoVe-M	CoVe-L	Char+CoVe-L
SST-2	84.2	88.4	90.1	89.0	90.9	91.1	**91.2**
SST-5	48.6	53.5	52.2	54.0	54.7	54.5	**55.2**
IMDb	88.4	91.1	91.3	90.6	91.6	91.7	**92.1**
TREC-6	88.9	94.9	94.7	94.7	95.1	95.8	**95.8**
TREC-50	81.9	89.2	89.8	89.6	89.6	90.5	**91.2**
SNLI	82.3	87.7	87.7	87.3	87.5	87.9	**88.1**
SQuAD	65.4	76.0	78.1	76.5	77.1	79.5	**79.9**

Pretrain bidirectional encoder with MT supervision, extract LSTM states

Adding CoVe with GloVe gives improvements for classification, NLI, Q&A

(McCann et al, NeurIPS 2017)

ELMo



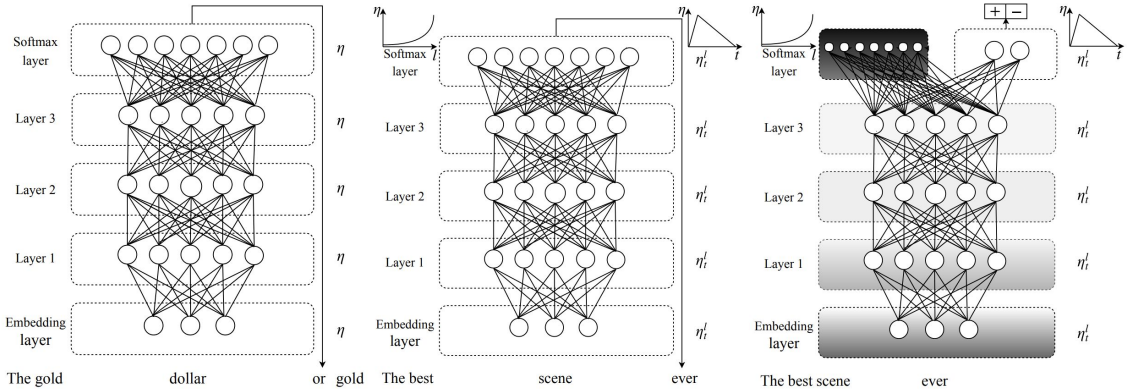
Task	Previous SOTA		Our baseline	ELMo + baseline	Increase (absolute/ relative)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al. (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%

Pretrain deep bidirectional LM, extract contextual word vectors as learned linear combination of hidden states

SOTA for 6 diverse tasks

(Peters et al, NAACL 2018)

ULMFiT



(a) LM pre-training (b) LM fine-tuning (c) Classifier fine-tuning

Pretrain AWD-LSTM LM,
fine-tune LM in two stages with
different adaptation techniques

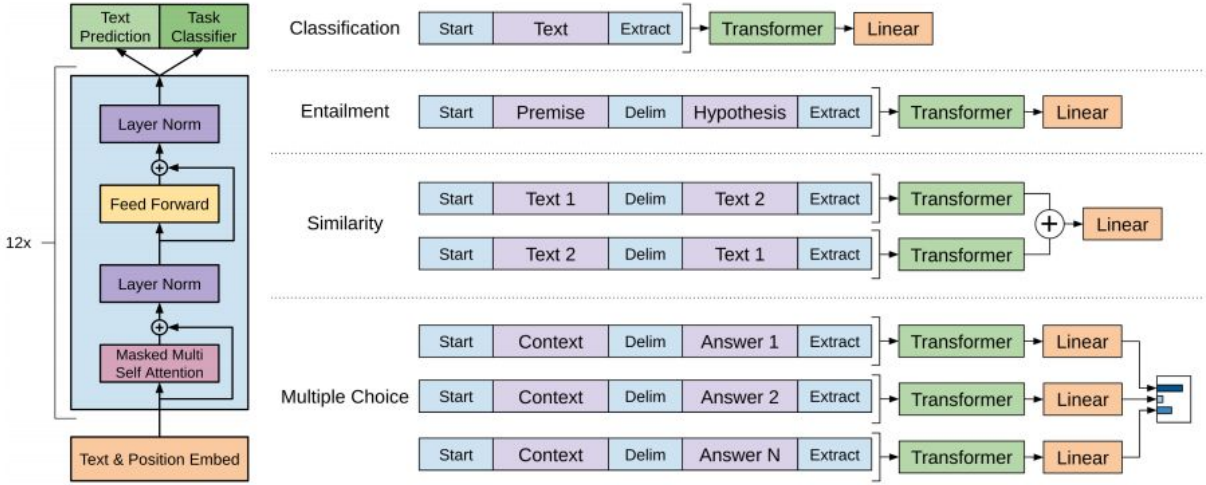
	Model	Test		Model	Test
IMDb	CoVe (McCann et al., 2017)	8.2	TREC-6	CoVe (McCann et al., 2017)	4.2
	oh-LSTM (Johnson and Zhang, 2016)	5.9		TBCNN (Mou et al., 2015)	4.0
	Virtual (Miyato et al., 2016)	5.9		LSTM-CNN (Zhou et al., 2016)	3.9
	ULMFiT (ours)	**4.6**		ULMFiT (ours)	**3.6**

SOTA for six classification
datasets

	AG	DBpedia	Yelp-bi	Yelp-full
Char-level CNN (Zhang et al., 2015)	9.51	1.55	4.88	37.95
CNN (Johnson and Zhang, 2016)	6.57	0.84	2.90	32.39
DPCNN (Johnson and Zhang, 2017)	6.87	0.88	2.64	30.58
ULMFiT (ours)	**5.01**	**0.80**	**2.16**	**29.98**

(Howard and Ruder, ACL 2018)

GPT



Pretrain large 12-layer left-to-right Transformer, fine tune for sentence, sentence pair and multiple choice questions.

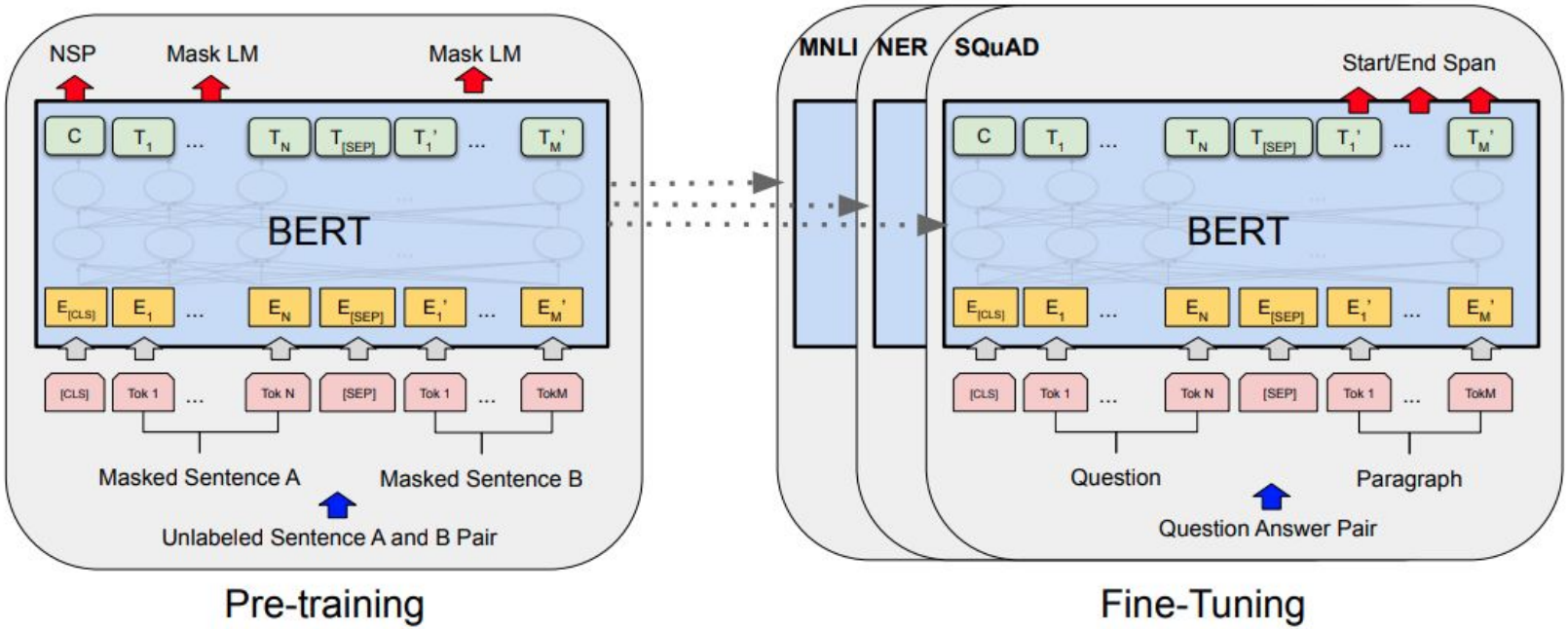
SOTA results for 9 tasks.

Method	MNLI-m	MNLI-mm	SNLI	SciTail	QNLI	RTE
ESIM + ELMo [44] (5x)	-	-	89.3	-	-	-
CAFE [58] (5x)	80.2	79.0	89.3	-	-	-
Stochastic Answer Network [35] (3x)	80.6	80.1	-	-	-	-
CAFE [58]	78.7	77.9	88.5	83.3		
GenSen [64]	71.4	71.3	-	-	82.3	59.2
Multi-task BiLSTM + Attn [64]	72.2	72.1	-	-	82.1	**61.7**
Finetuned Transformer LM (ours)	**82.1**	**81.4**	**89.9**	**88.3**	**88.1**	56.0

(Radford et al., 2018)

BERT

BERT pretrains both sentence and contextual word representations,
using masked LM and next sentence prediction.
BERT-large has 340M parameters, 24 layers!



See also: [Logeswaran and Lee, ICLR 2018]

[(Devlin et al. 2019)]

BERT

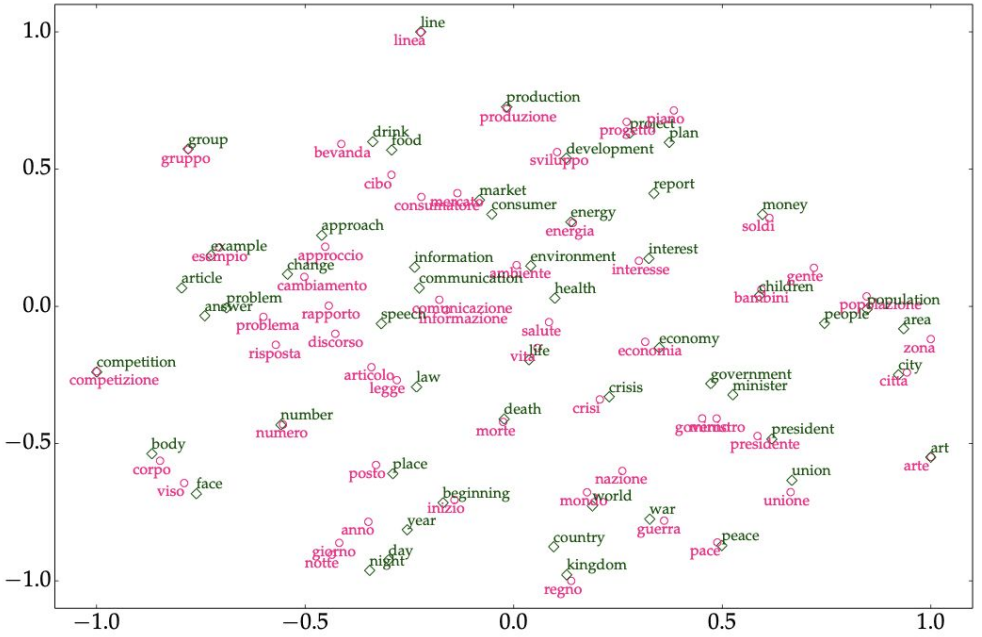
SOTA GLUE benchmark results (sentence pair classification).

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	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

(Devlin et al. 2019)

Cross-lingual pretraining

- Much work on training cross-lingual word embeddings (Overview: Ruder et al. (2017))
- Idea: train each language separately, then align.
- Recent work aligning ELMo: Schuster et al., (NAACL 2019)
- ACL 2019 Tutorial on Unsupervised Cross-lingual Representation Learning



Cross-lingual pretraining

Cross-lingual Polyglot Pretraining

Key idea: **Share vocabulary** and representations across languages by training one model on many languages.

Advantages: Easy to implement, **enables** cross-lingual pretraining by itself

Disadvantages: Leads to **under-representation** of low-resource languages

Artetxe & Schwenk, 2018
Multilingual BERT

Mulcaire et al., NAACL 2019

Lample & Conneau, 2019

Data resources for NLP

- https://huggingface.co/datasets
-

Best practice of learning and understanding Deep Learning

Step 1 - Build-your-own Neural Network Toolkit: Write the logic from the scratch

Step 2 - Implement your NN in any NLP application

- Start with simple Text Classifier/predictor: predict next word

Step 3 - SOTA Survey / Re-implementation: Re- implement and reproduce results from a recently published NLP paper

Step 4 - Perform a unique project that either (1) improves on state-of-the-art, or (2) applies NLP models to a unique task.

Step 5: Write good article on it

Reflection

Are we doing right thing to deal with nlp problems?

Common formula

-

Useful Resources

https://nlp.seas.harvard.edu/2018/04/03/attention.html

https://github.com/graykode/nlp-tutorial

- wm2wts@gmail.com
- abebawu.eshetu@adludio.com
- Linkedin: Abebaw

Thank you!