The ImageNet Moment in NLP: From wordvectors to language models

Tobias Sterbak

Outline:

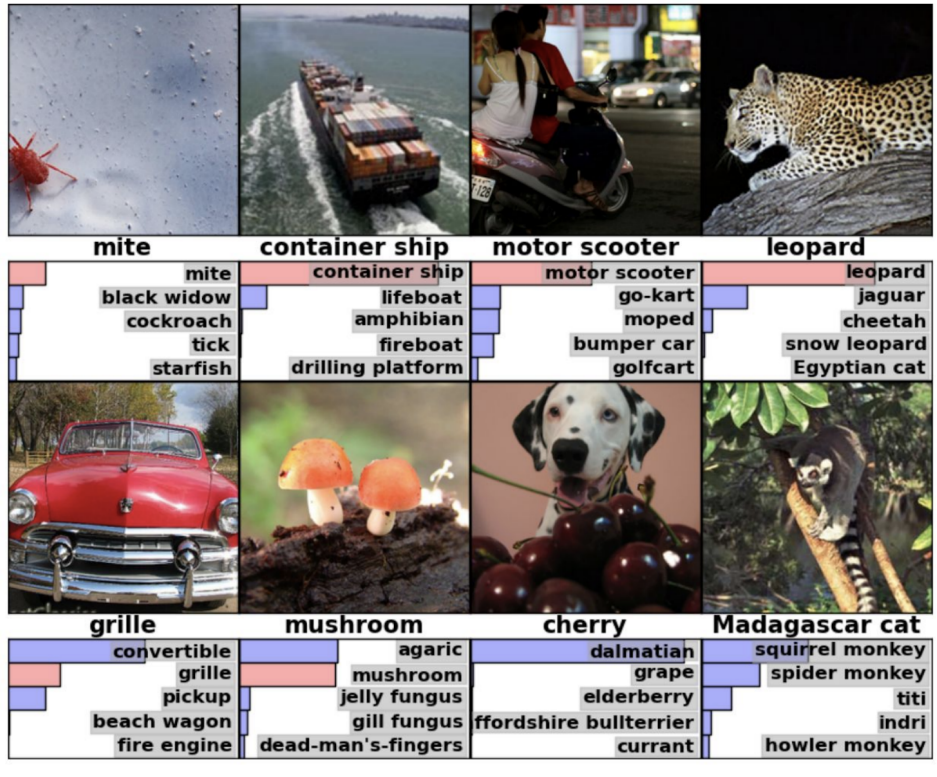
1. Why is everyone so exited about transfer learning?
2. Recap wordvectors
3. Language models
4. Feature-based methods
5. Fine-tuning based methods
6. Words of warning

1. Why is everyone so exited about transfer learning?

- Labeled data is sparse
- Training deep neural networks is expensive
- Strong performance

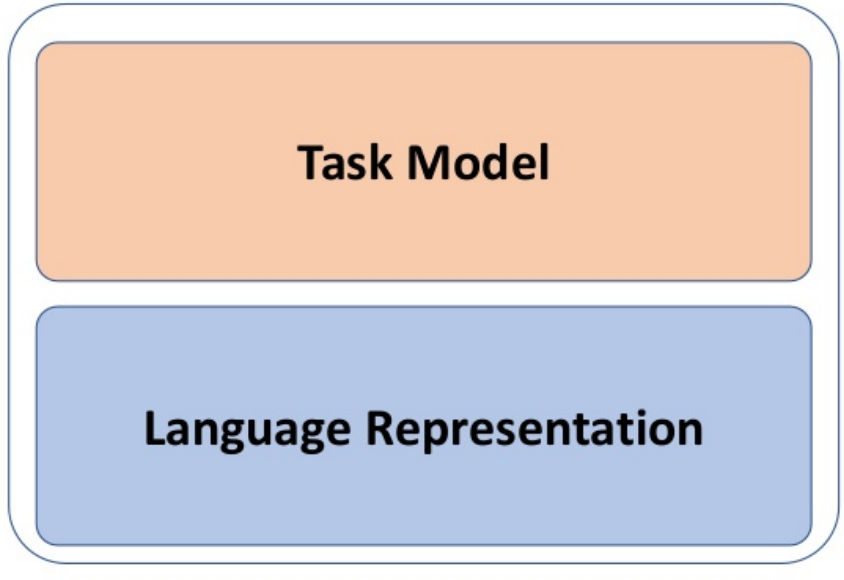
What is ImageNet?

ImageNet Challenge



- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.

What about NLP?



Two components of an NLP model

*Image from: Comparison of Transfer-Learning Approaches for Response Selection in Multi-Turn Conversations, Jesse Vig, Kalai Ramea, DSTC 7 Workshop, AAAI'19 Honolulu

Traditional approach

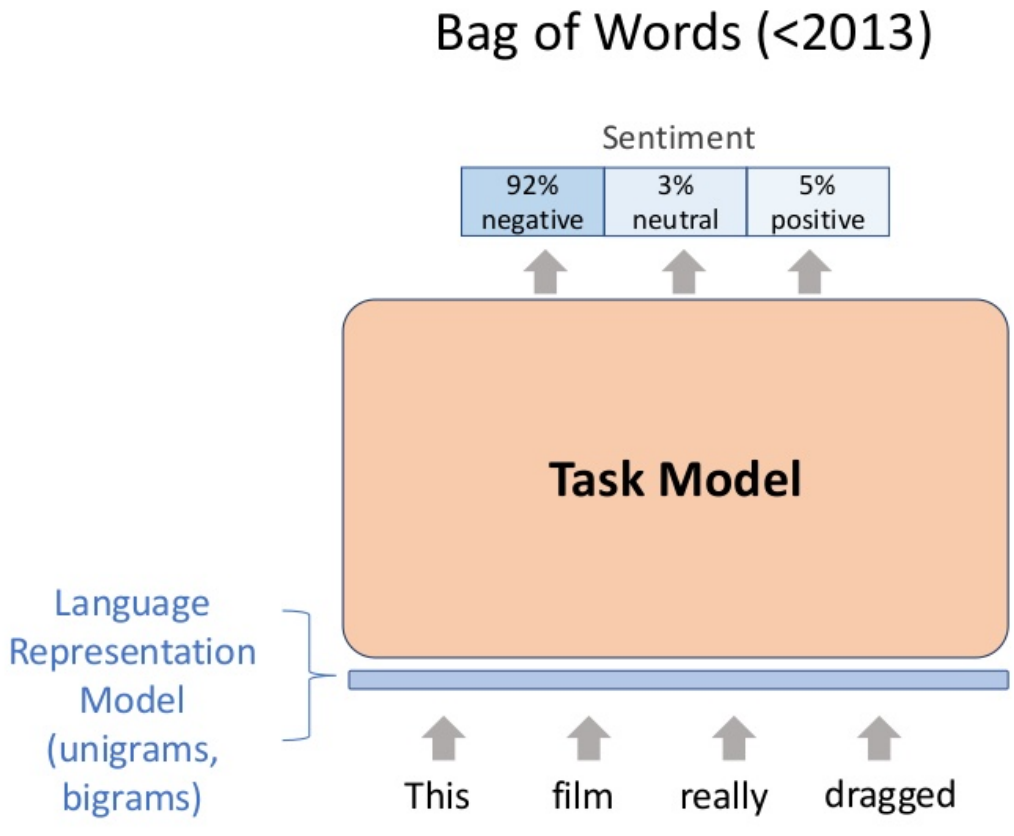
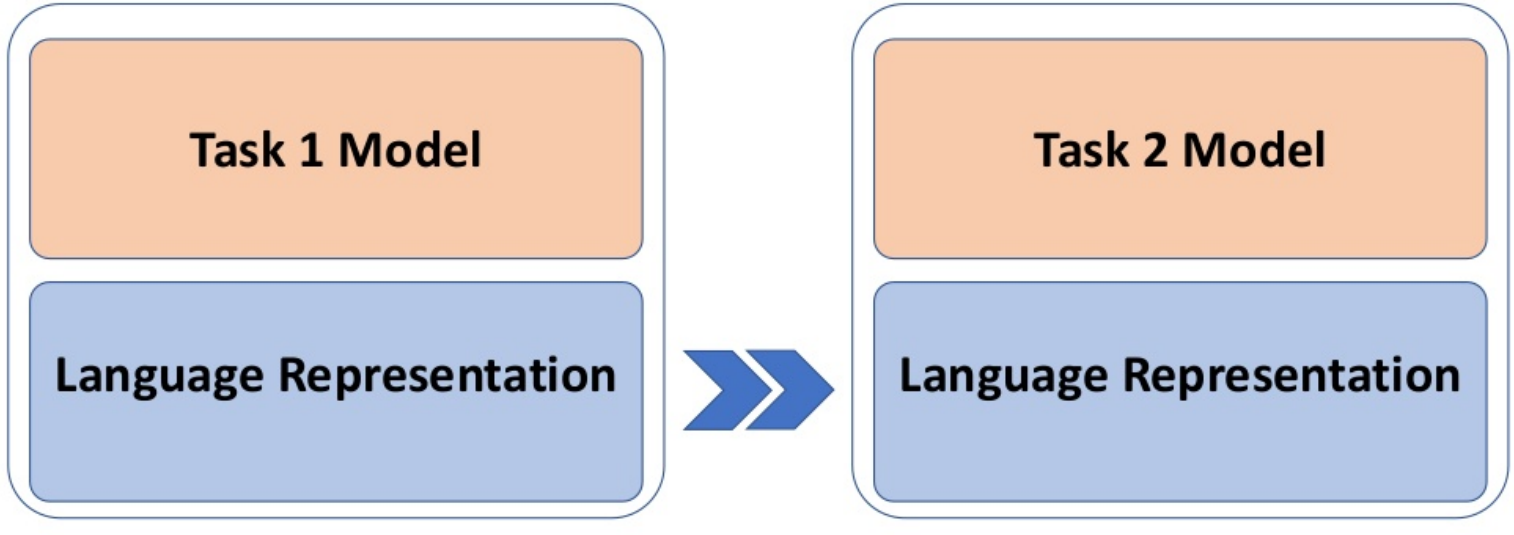


*Image from: Comparison of Transfer-Learning Approaches for Response Selection in Multi-Turn Conversations, Jesse Vig, Kalai Ramea, DSTC 7 Workshop, AAAI'19 Honolulu

Main Idea



Language representation is common across tasks, enabling transfer learning.

*Image from: Comparison of Transfer-Learning Approaches for Response Selection in Multi-Turn Conversations, Jesse Vig, Kalai Ramea, DSTC 7 Workshop, AAAI'19 Honolulu

2. Recap wordvectors:

Idea: Capture the meaning of a word in a vector.

Distibutional hypothesis:

"Words that appear in the same context have similar meaning."
Harris, 1954

Word embeddings (2013-)

Word2Vec / GloVe / FastText

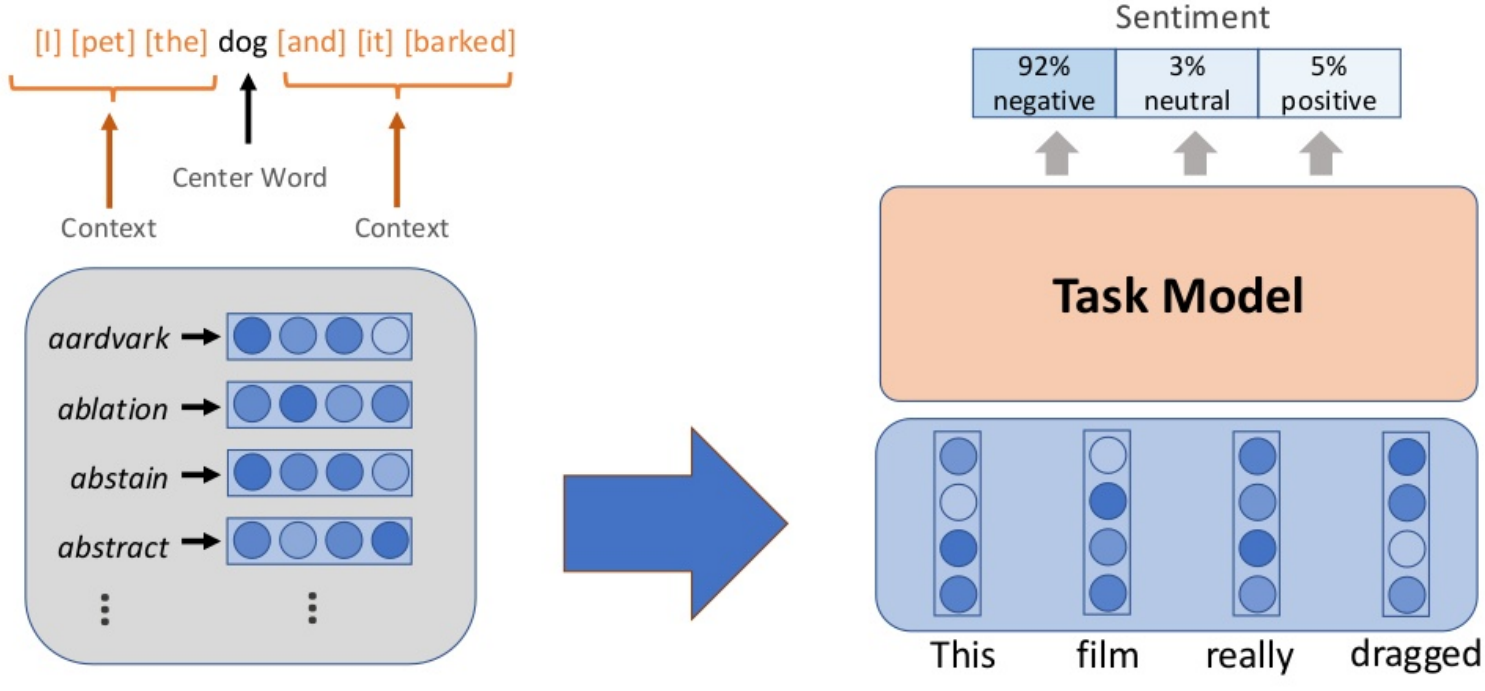


*Image from: Comparison of Transfer-Learning Approaches for Response Selection in Multi-Turn Conversations, Jesse Vig, Kalai Ramea, DSTC 7 Workshop, AAAI'19 Honolulu

Quite Succesfull!

But probably we're missing something?

Problem: The same words can have multiple meanings depending on the context!

- This is not an **Apple** ad!
- Why Is New York City called „The Big **Apple**“?
- I like to eat an **apple**.

So **CONTEXT** matters!

3. Language models

"Predict the next token given the previous tokens."

I want a ...

p(beer | I, want, a) = ?

p(pizza | I, want, a) = ?

3. Language models

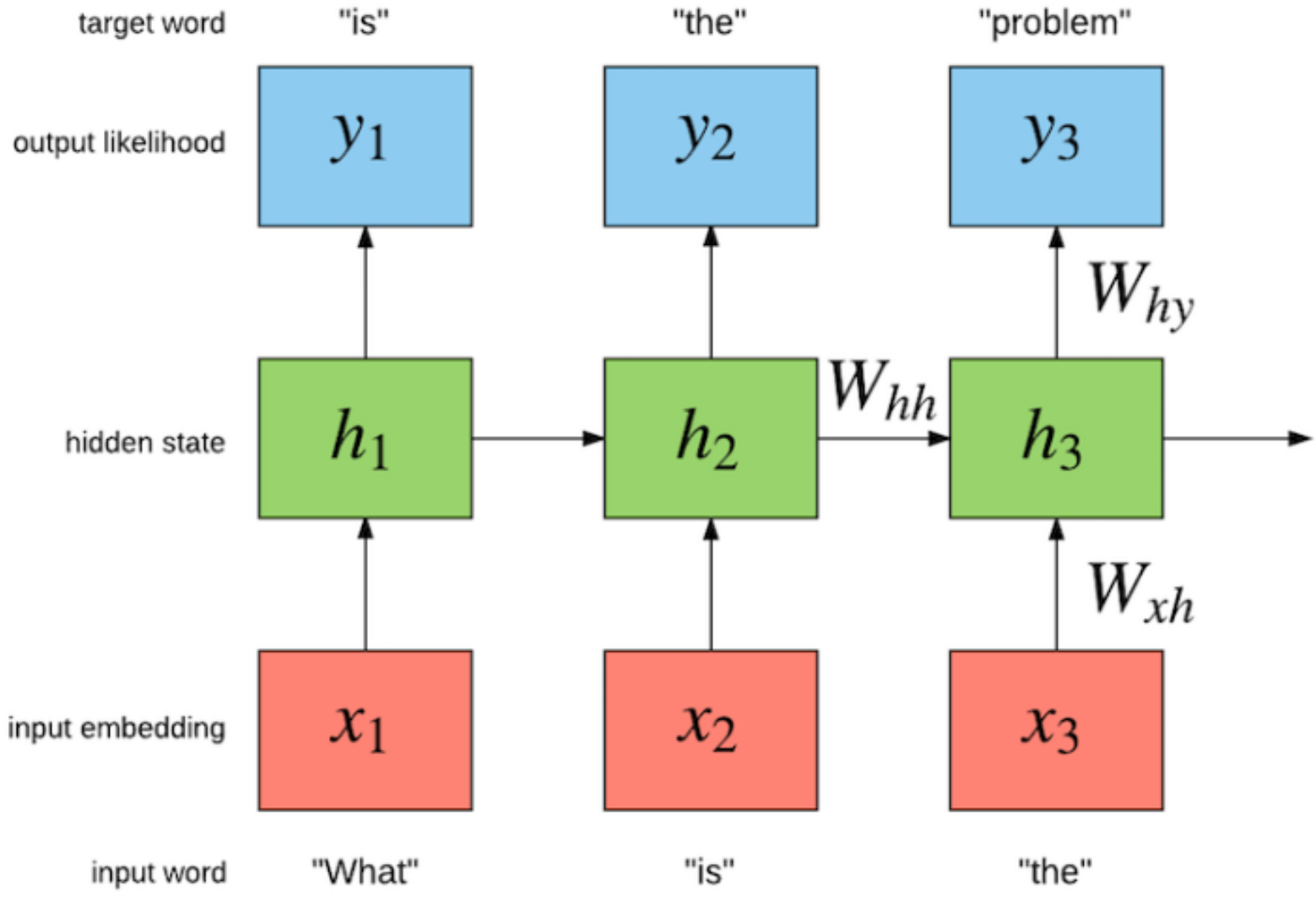


*Image from: http://torch.ch/blog/2016/07/25/nce.html

4. Feature-based methods

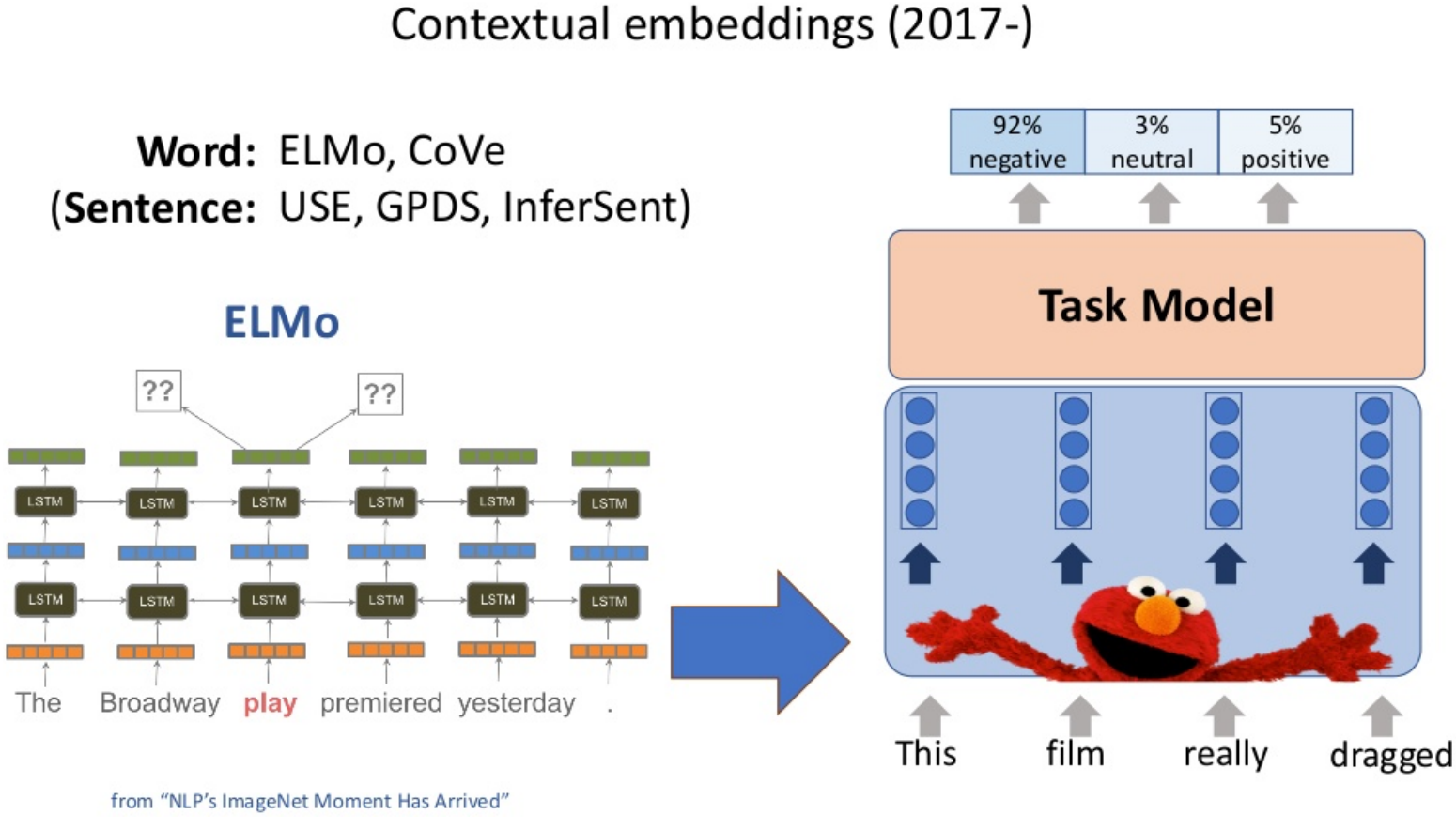


*Image from: Comparison of Transfer-Learning Approaches for Response Selection in Multi-Turn Conversations, Jesse Vig, Kalai Ramea, DSTC 7 Workshop, AAAI'19 Honolulu

4. Feature-based methods

Pros:

- Requires less resources than fine-tuning

Cons:

- Requires a customized model for each downstream task
- (Generally) Scores lower than fine-tuning

ELMo with tensorflow

```
import tensorflow as tf
import tensorflow_hub as hub

embedded_text_feature_column = hub.text_embedding_column(
    key="sentence",
    module_spec="https://tfhub.dev/google/elmo/2")

estimator = tf.estimator.DNNClassifier(
    hidden_units=[500, 100], n_classes=2,
    feature_columns=[embedded_text_feature_column],
    optimizer=tf.train.AdagradOptimizer(learning_rate=0.003))

estimator.train(input_fn=train_input_fn, steps=10000)
```

ELMo with keras

```
from keras.models import Input
from keras.layers import Lambda
import tensorflow as tf
import tensorflow_hub as hub

elmo = hub.Module("https://tfhub.dev/google/elmo/2", trainable=True)

def ELMoEmbedding(x):
    return elmo(tf.squeeze(tf.cast(x, tf.string)),
                signature="default",
                as_dict=True)["default"]
input_text = Input(shape=(1,), dtype=tf.string)
embedding = Lambda(ELMoEmbedding, output_shape=(None, 1024))(input_text)
```

5. Fine-tuning-based methods

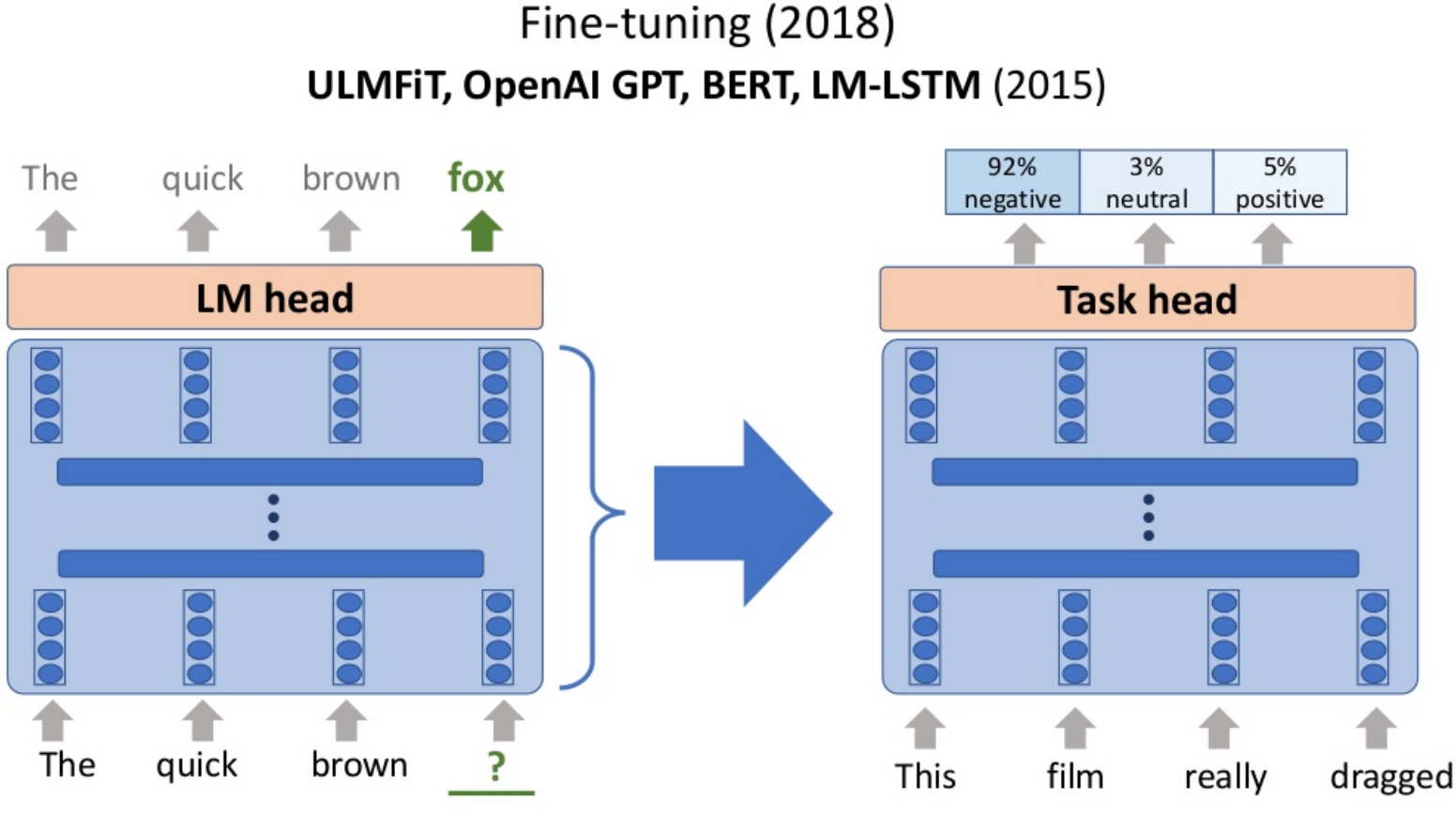


*Image from: Comparison of Transfer-Learning Approaches for Response Selection in Multi-Turn Conversations, Jesse Vig, Kalai Ramea, DSTC 7 Workshop, AAAI'19 Honolulu

5. Fine-tuning-based methods

Pros:

- Simple structure
- (Generally) Scores higher than feature-based

Cons:

- Requires more resources due to re-training pre-trained models that are originally big
- Larger latency is expected when serving (as a trade-off for higher score)

BERT with pytorch

```
from pytorch_pretrained_bert import BertTokenizer, BertForSequenceClassification

tokenizer = BertTokenizer.from_pretrained('bert-large-uncased')
model = BertForSequenceClassification.from_pretrained('bert-large-uncased')
```

ULMFiT with fastai

```
from fastai import *
from fastai.text import *

data_lm = TextLMDataBunch.from_df(train_df=df_trn, valid_df=df_val, path="")
data_clas = TextClasDataBunch.from_df(path="", train_df=df_trn, valid_df=df_val,
vocab=data_lm.train_ds.vocab, bs=32)

learn = language_model_learner(data_lm)

learn.fit_one_cycle(1, lr=1e-2)
learn.save_encoder('lm_enc')

clf = text_classifier_learner(data_clas)
clf.load_encoder('lm_enc')
learn.fit_one_cycle(1, lr=1e-2)
```

6. Words of warning

- Try simple baselines first!
- Know your data!
- Language models learn dataset bias!

Resources

Theory:

- https://lilianweng.github.io/lil-log/2019/01/31/generalized-language-models.html (https://lilianweng.github.io/lil-log/2019/01/31/generalized-language-models.html)
- http://www.davidsbatista.net/blog/2018/12/06/Word_Embeddings/ (http://www.davidsbatista.net/blog/2018/12/06/Word_Embeddings/)
- https://www.slideshare.net/jessevig5/comparison-of-transferlearning-approaches-for-response-selection-in-multiturn-conversations-130633835 (https://www.slideshare.net/jessevig5/comparison-of-transferlearning-approaches-for-response-selection-in-multiturn-conversations-130633835)

Code:

Where to find me

www.depends-on-the-definition.com

www.github.com/tsterbak

@tobias_sterbak