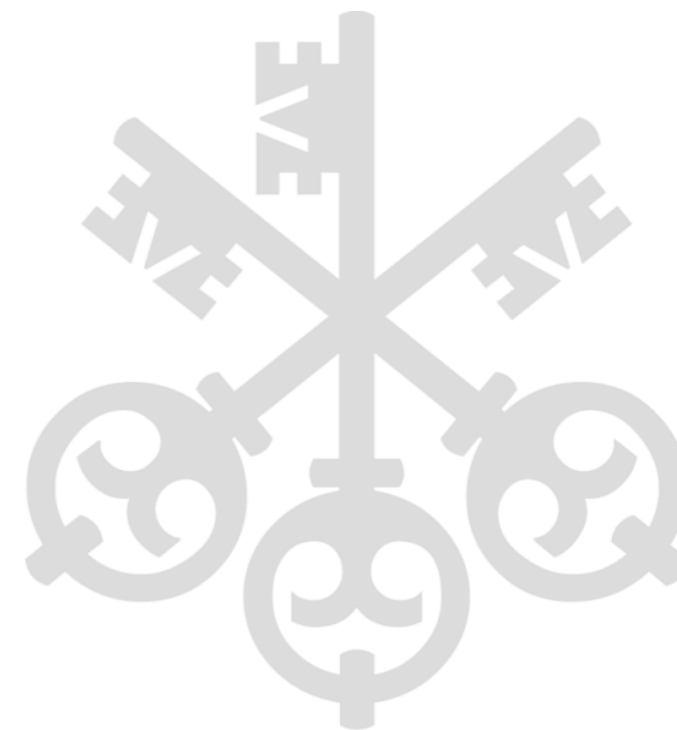


# Deep (Transfer) Learning for NLP on Small Data Sets

Evaluating efficacy and application of techniques

Hanoz Bhatena and Raghav 'Mady' Madhavan  
UBS Evidence Lab

March 20, 2019





## Disclaimer

Opinions and views shared here are our personal ones, and not those of UBS or UBS Evidence Lab.

Any mention of Companies, Public or Private, and/or their Brands, Products or Services is for illustrative purposes only and does not reflect a recommendation.

# Agenda

---

- Problem & Motivation
- Transfer Learning Fundamentals
- Transfer Learning for small datasets in NLP
- Experiments
- Results
- Conclusion
- Future Work
- Q & A

# Problem

---

- Large (labeled) datasets has been the fuel that has powered the deep learning revolution of NLP
- However, in common business contexts, labeled data can be scarce
- Examples:
  - Financial documents
  - Legal documents
  - Client feedback emails
  - Classification from Clinical visits
- Issues:
  - Expensive to get labeling services
  - Data privacy concerns
  - Experimentation phase (unknown payoff; when to stop tagging?)

# Motivation

---

Enable building deep learning models when small quantities of labeled data are available



Increase usability of deep learning for NLP tasks



Decrease time required to develop models



Democratize model development beyond NLP experts

# Deep learning with less labeled data

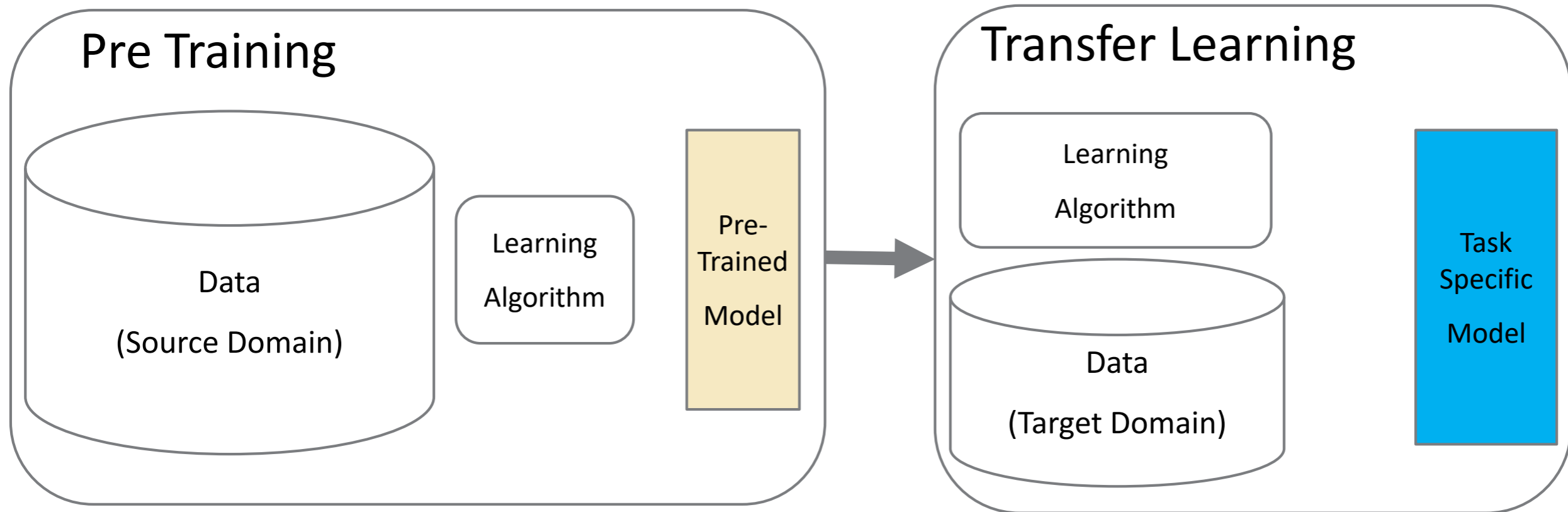
---

- Transfer learning
- Semi-supervised learning
- Artificial data augmentation
- Weak supervision
- Zero-shot learning
- One-shot learning
- Few shot learning
- .....

# Deep Transfer Learning Introduction

Use a model trained for one or more tasks

to solve another different, but somewhat related, task



*After supervised learning — Transfer Learning will be the next driver of ML commercial success - Andrew Ng, NIPS 2016*

# Transfer Learning in Computer Vision

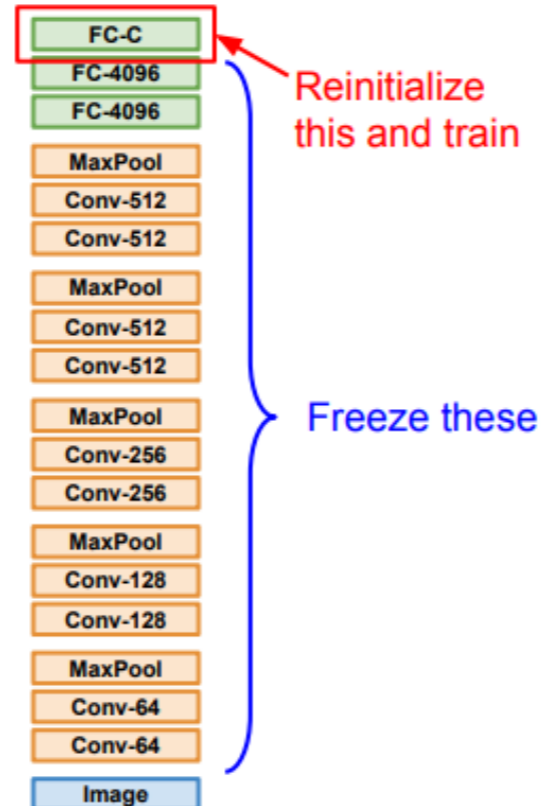
## Transfer Learning with CNNs

Donahue et al, "DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition", ICML 2014  
Razavian et al, "CNN Features Off-the-Shelf: An Astounding Baseline for Recognition", CVPR Workshops 2014

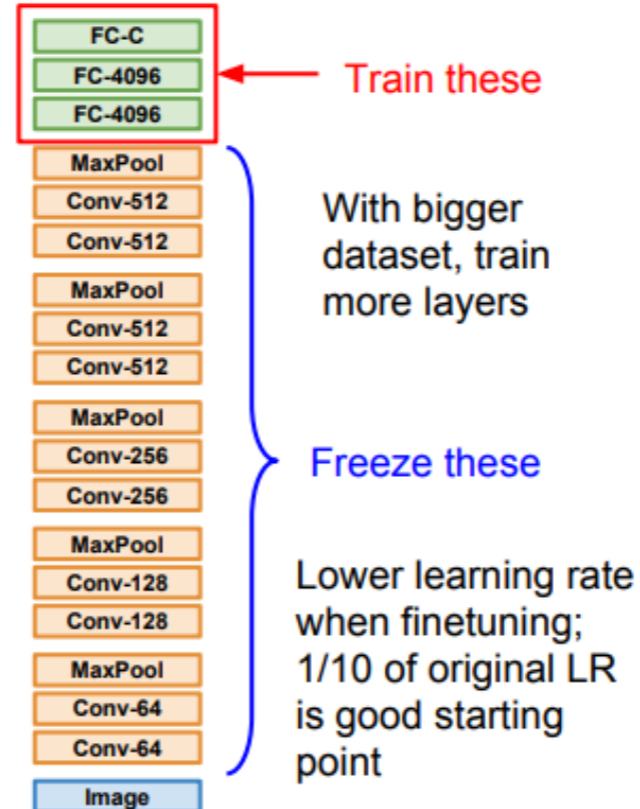
### 1. Train on Imagenet



### 2. Small Dataset (C classes)



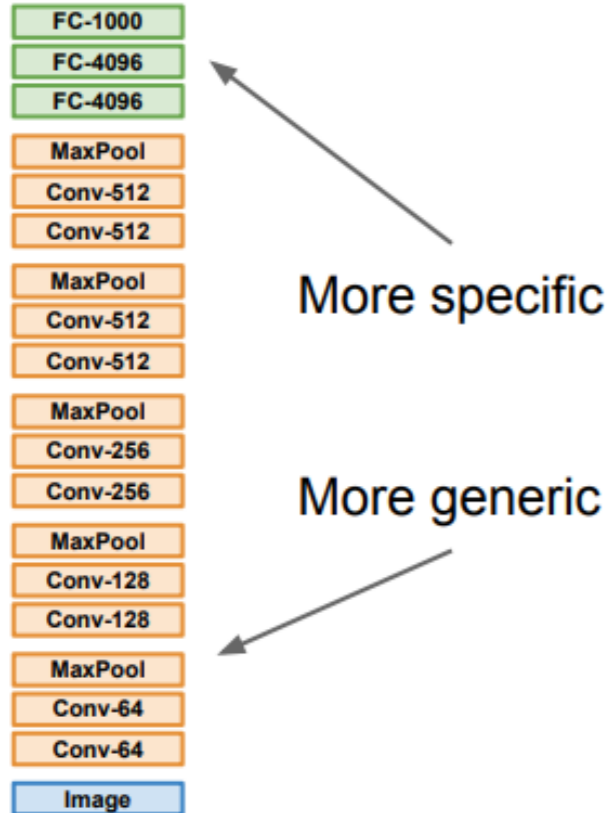
### 3. Bigger dataset



Source: Stanford CS231N lecture slides: Fei-Fei Li & Justin Johnson & Serena Yeung



# Transfer Learning – General Rule



	<b>very similar dataset</b>	<b>very different dataset</b>
<b>very little data</b>	Use Linear Classifier on top layer	You're in trouble... Try linear classifier from different stages
<b>quite a lot of data</b>	Finetune a few layers	Finetune a larger number of layers

Source: Stanford CS231N lecture slides: Fei-Fei Li & Justin Johnson & Serena Yeung

# So, what about Transfer Learning for NLP?

---

- Is there a source dataset like ImageNet for NLP?
- Does this dataset require annotations? Or can we leverage unsupervised learning somehow?
- What are some common model architectures for NLP problems that optimize for knowledge transfer?
- How low can we go in terms of data requirements in our target domain?
- Should we tune the entire pre-trained model or just use it as a feature generator for downstream tasks?

# Transfer Learning for NLP – Pre-2018

---

- Word2Vec (Feature based and Fine-tunable) (<https://arxiv.org/abs/1310.4546>)
- Glove (Feature based and Fine-tunable) (<https://nlp.stanford.edu/pubs/glove.pdf>)
- FastText (Feature based and Fine-tunable) (<https://arxiv.org/abs/1607.04606>)
- Sequence Autoencoders (Feature based and Fine-tunable) (<https://arxiv.org/abs/1511.01432>)
- LSTM language model pre-training (Feature based and Fine-tunable) (<https://arxiv.org/abs/1511.01432>)

# Transfer Learning for NLP – 2018 and Beyond

---

- Supervised Learning of Universal Sentence Representations from NLI Data (InferSent) (<https://arxiv.org/abs/1705.02364>) \*\*
- Deep contextualized word representations (ELMo) (<https://arxiv.org/abs/1802.05365>)
- Universal Sentence Encoder (<https://arxiv.org/abs/1803.11175>)
- OpenAI GPT ([https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language\\_understanding\\_paper.pdf](https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf))
- BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (<https://arxiv.org/abs/1810.04805>)
- Universal Language Model Fine-tuning for Text Classification (ULMFiT) (<https://arxiv.org/abs/1801.06146>)
- GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding (<https://arxiv.org/abs/1804.07461>, <https://github.com/nyu-mll/GLUE-baselines>)
- OpenAI GPT 2 ([https://d4mucfpksywv.cloudfront.net/better-language-models/language\\_models\\_are\\_unsupervised\\_multitask\\_learners.pdf](https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf))

\*\* This was actually published in 2017

# What is GLUE and how is our objective different?

Corpus	Train	Dev	Test	Task	Metrics	Domain
Single-Sentence Tasks						
CoLA	8.5k	1k	<b>1k</b>	acceptability	Matthews corr.	misc.
SST-2	67k	872	1.8k	sentiment	acc.	movie reviews
Similarity and Paraphrase Tasks						
MRPC	3.7k	408	1.7k	paraphrase	acc./F1	news
STS-B	7k	1.5k	1.4k	sentence similarity	Pearson/Spearman corr.	misc.
QQP	364k	40k	<b>391k</b>	paraphrase	acc./F1	social QA questions
Inference Tasks						
MNLI	393k	20k	<b>20k</b>	NLI	matched acc./mismatched acc.	misc.
QNLI	108k	5.7k	5.7k	QA/NLI	acc.	Wikipedia
RTE	2.5k	276	3k	NLI	acc.	misc.
WNLI	634	71	<b>146</b>	coreference/NLI	acc.	fiction books

Table 1: Task descriptions and statistics. All tasks are single sentence or sentence pair classification, except STS-B, which is a regression task. MNLI has three classes; all other classification tasks have two. Test sets shown in bold use labels that have never been made public in any form.

Source: Original GLUE paper (<https://arxiv.org/abs/1804.07461>)

- Because with exception of WNLI (and perhaps RTE), most of these datasets are still too large to create especially for experimental projects in a commercial setting.
- Is it possible to create meaningful deep learning models for classification on just a few hundred samples?

# Deep contextualized word representations (ELMo)

---

- Generates context dependent word embeddings
- Example: the word vector for the word "bank" in the sentence "I am going to the bank" will be different from the vector for the sentence "We can bank on him"
- The model comprises of a character level CNN model followed by a L=2 layer bi-directional LSTM model
- Weighted average of the embeddings from char-CNN and the hidden vectors from the 2 layer bi-LSTM
- Language model pretraining on the 1B Word Benchmark
- Pre-trained model is available on Tensorflow-Hub and AllenNLP

# Universal Sentence Encoder

---

- Two types: Deep Averaging Network (DAN) and Transformer network
- Multi-task training on a combination of supervised and unsupervised training objectives
- Trained on varied datasets like Wikipedia, web news, blogs
- Uses attention to compute context aware word embeddings which are combined into a sentence level representation
- Pre-trained model is available on Tensorflow-Hub

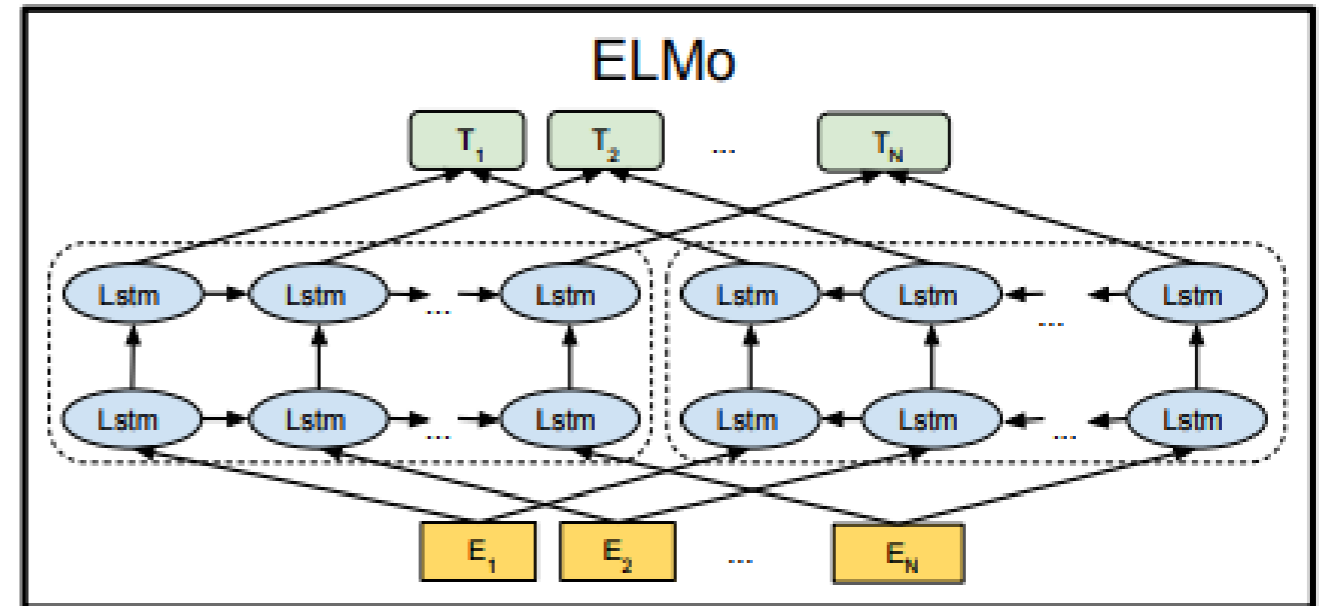
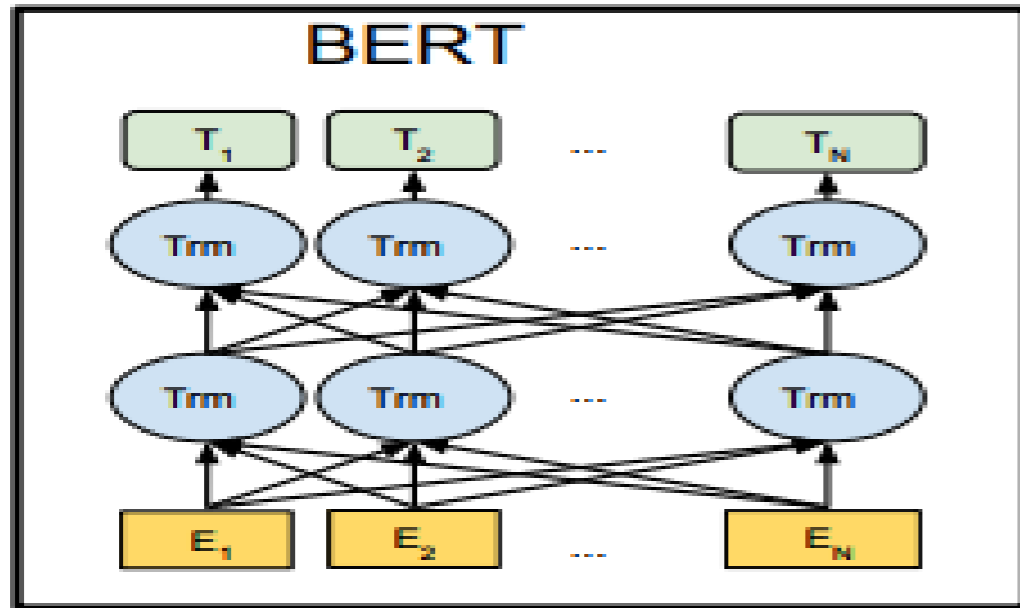
# BERT

---

- Uses the encoder half of Transformer
- The input is tokenized using a WordPiece tokenizer (Wu et al., 2016)
- Training on a dual task: Masked LM and next sentence prediction
- The next sentence prediction task learns to predict, given two sentences A and B, whether the second sentence (B) comes after the first one (A)
- This enables the BERT model to understand sentence relationships and thereby a higher level understanding capability compared to just a language model training
- Data for pre-training: BookCorpus (800mn words) + English Wikipedia (2.5bn words)
- BERT obtains SOTA results on 11 NLP tasks in the GLUE benchmark



# BERT vs ELMo - Architecture



Source: Original BERT paper

# Experiments: Setup

---

- Feature based learning: Only train the final layer(s)
  - Finetune based learning: Fine tune all layers using a small learning rate
- } Transfer learning training paradigms
- Baseline CNN (with and without pretrained Glove embeddings)
  - ELMo
  - Universal Sentence Encoder
  - BERT
- } Models to evaluate
- Mean, Standard Deviation of Out-of-Sample Accuracy after N trials
  - No explicit attempt to optimize hyperparameters
- } Evaluation Criteria
- Some pre-trained model architecture will be well suited for all applications
  - Either finetuning or feature mode will emerge a consistent winner
- } Apriori Expectations

# Experiment 1: IMDB Rating Application

---

- Sentiment classification model on IMDB movie reviews
- Binary classification problem: positive or negative
- 25,000 Training samples; 12,500 positive and 12,500 negative
- 25,000 Test samples; 12,500 positive and 12,500 negative

★ 9/10


**Nitpicking is only criticism**  
19 November 2005

There is so little to find wanting in this film and not wanting to merely repeat its many deserved praises, all I can do is question a couple of historical points.

It is unlikely that a Roman general would be sold into slavery and forced to fight in the arena. Exiled, yes. Killed maybe. Asked to commit suicide to retain his property, most likely.

It is unlikely that he would return to find his family crucified, of all things. Romans were very specific about who got crucified and why. Romans usually avoided it, no matter how cruel the tyrant(?) was.

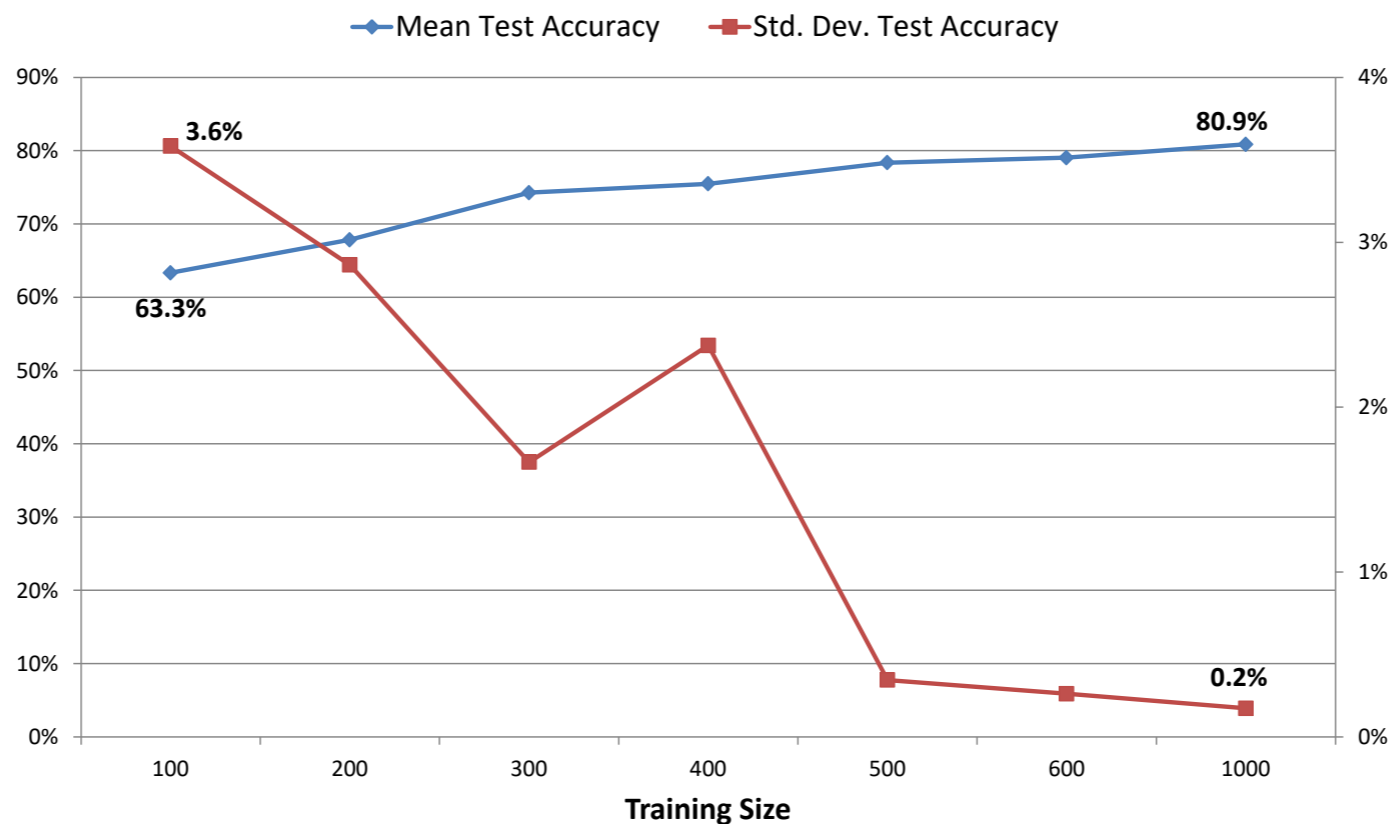
Roman legionnaires would NEVER have a tattoo unless they were barbarians who got one

26 out of 37 found this helpful. Was this review helpful? [Sign in](#) to vote. 

# Experiment 1: IMDB Rating Application

Naïve baseline model: CNN with BatchNorm and Dropout WITHOUT pretrained Glove

100 Trials each

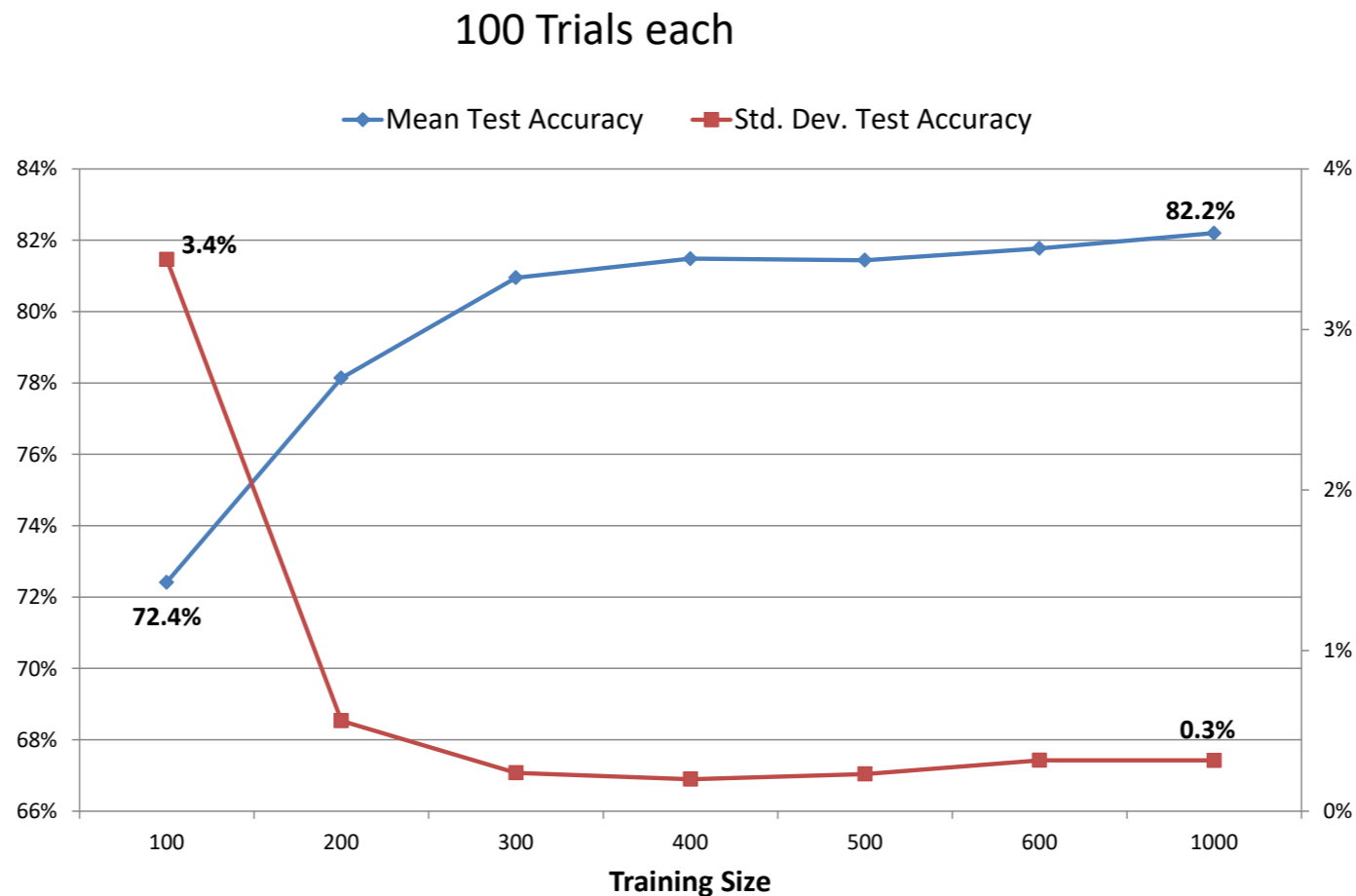


Using 25,000 training sample yields: 87.1%

Source: UBS Evidence Lab

# Experiment 1: IMDB Rating Application

More realistic baseline model: CNN with BatchNorm and Dropout WITH pretrained Glove



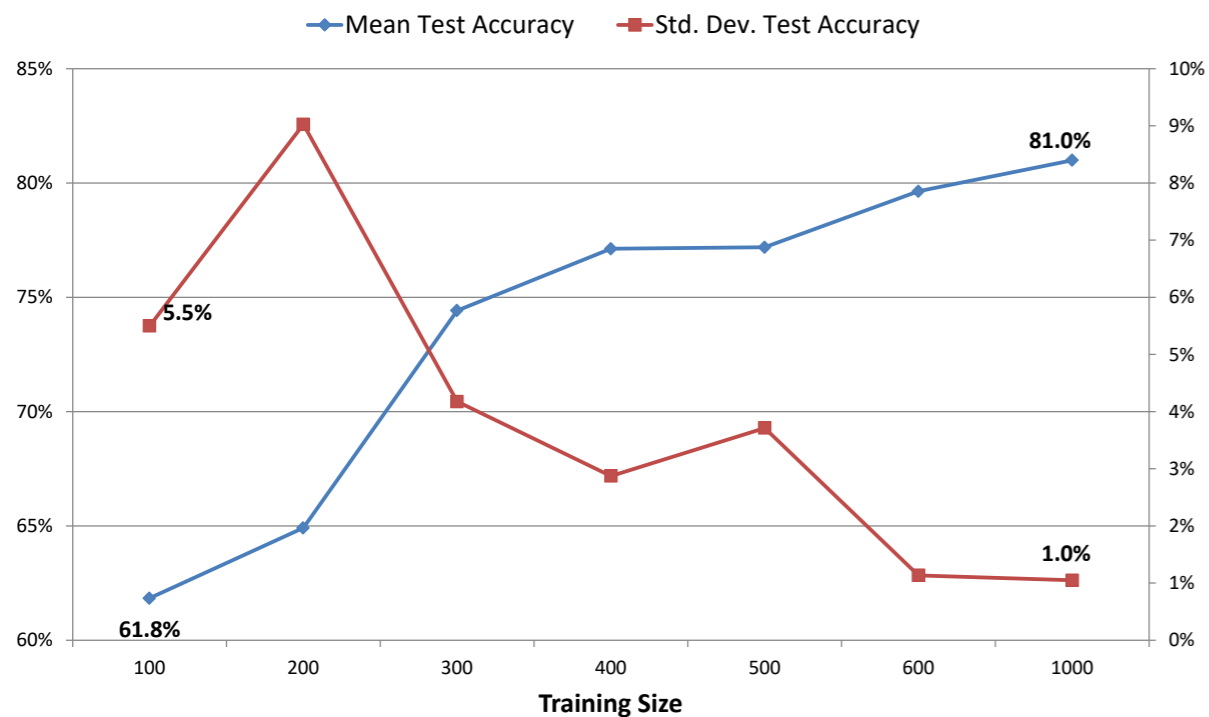
Using 25,000 training sample yields: 89.8%

Source: UBS Evidence Lab

# Experiment 1: IMDB Rating Application

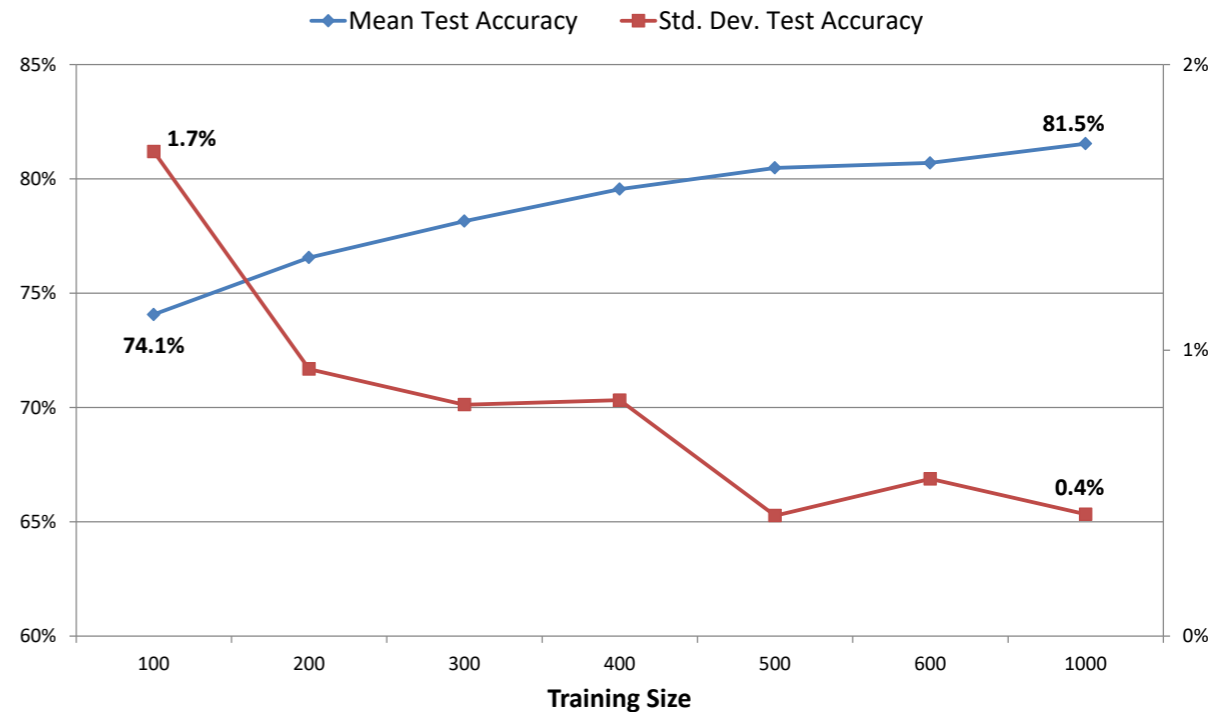
## Universal Sentence Encoder: DAN

### Fine Tuning based Training – 10 Trials each



Using 25,000 training sample yields: 86.6%

### Feature based Training – 10 Trials each



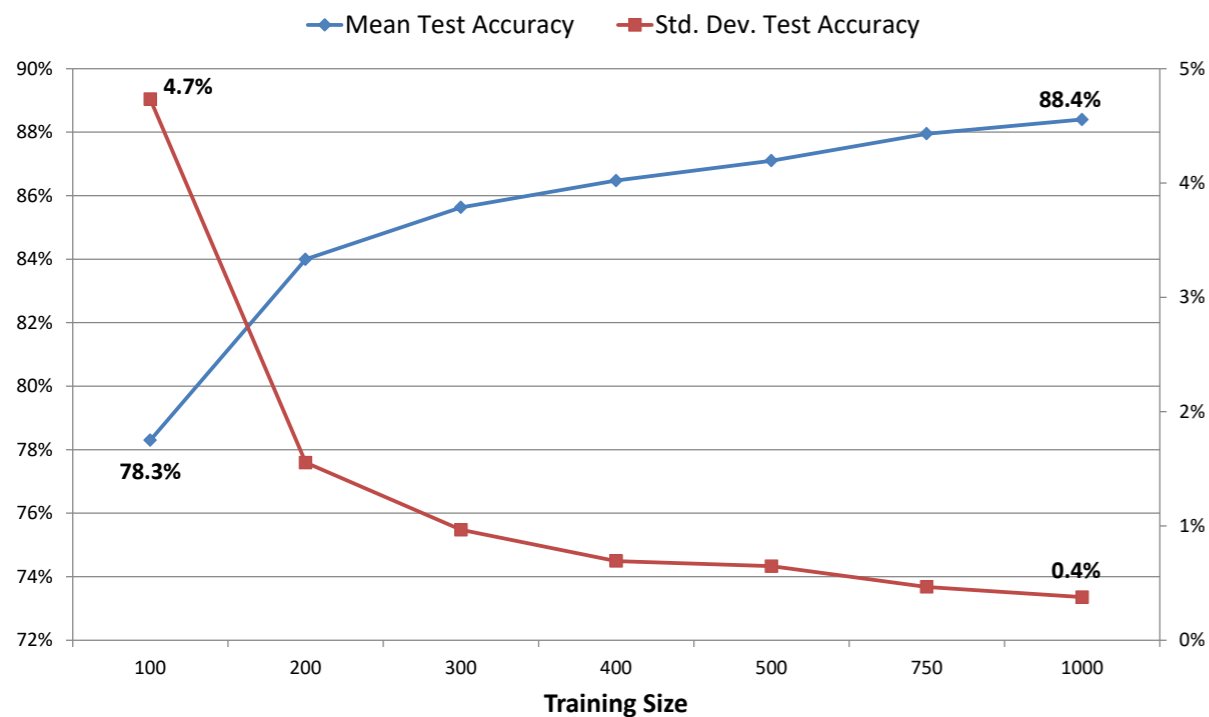
Using 25,000 training sample yields: 82.6%

Source: UBS Evidence Lab

# Experiment 1: IMDB Rating Application

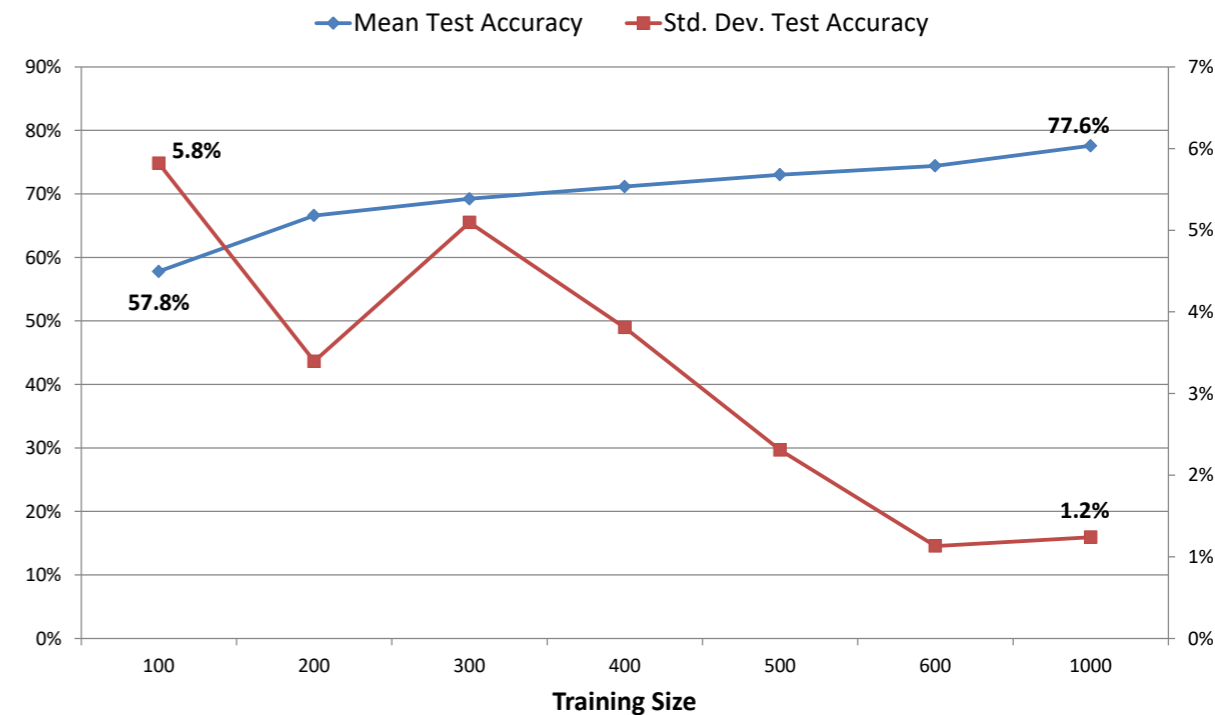
## BERT

### Fine Tuning based Training – 100 Trials each



Using 25,000 training sample yields: 92.5%

### Feature based Training – 10 Trials each



Using 25,000 training sample yields: 81.8%

Source: UBS Evidence Lab

# Experiment 1: IMDB Rating Application

## Summary of Experimental Results

<b>Model</b>	<b>100</b>	<b>200</b>	<b>300</b>	<b>400</b>	<b>500</b>	<b>600</b>	<b>1000</b>
<b>Naïve Baseline</b>	61%	66%	73%	74%	78%	79%	81%
<b>Realistic Baseline</b>	70%	78%	81%	81%	81%	82%	82%
<b>USE - FT</b>	59%	60%	71%	75%	74%	79%	80%
<b>USE - FB</b>	73%	76%	78%	79%	80%	80%	81%
<b>BERT - FT</b>	75%	83%	85%	86%	87%	88%	88%
<b>BERT - FB</b>	55%	64%	66%	69%	71%	74%	77%

$$\text{Adjusted Accuracy} = \frac{\text{Accuracy}}{(1 + \text{Stddev})}$$



# Experiment 2: HyperPartisan News Application

---

- Given a news article text, decide whether it follows a hyperpartisan argumentation, i.e., whether it exhibits blind, prejudiced, or unreasoning allegiance to one party, faction, cause, or person.  
(<https://pan.webis.de/semeval19/semeval19-web/>)
- Binary classification problem: Whether a news article is hyperpartisan or not
- 642 Training samples; 50% hyperpartisan and 50% neutral
- 129 Test samples; 67% hyperpartisan and 33% neutral

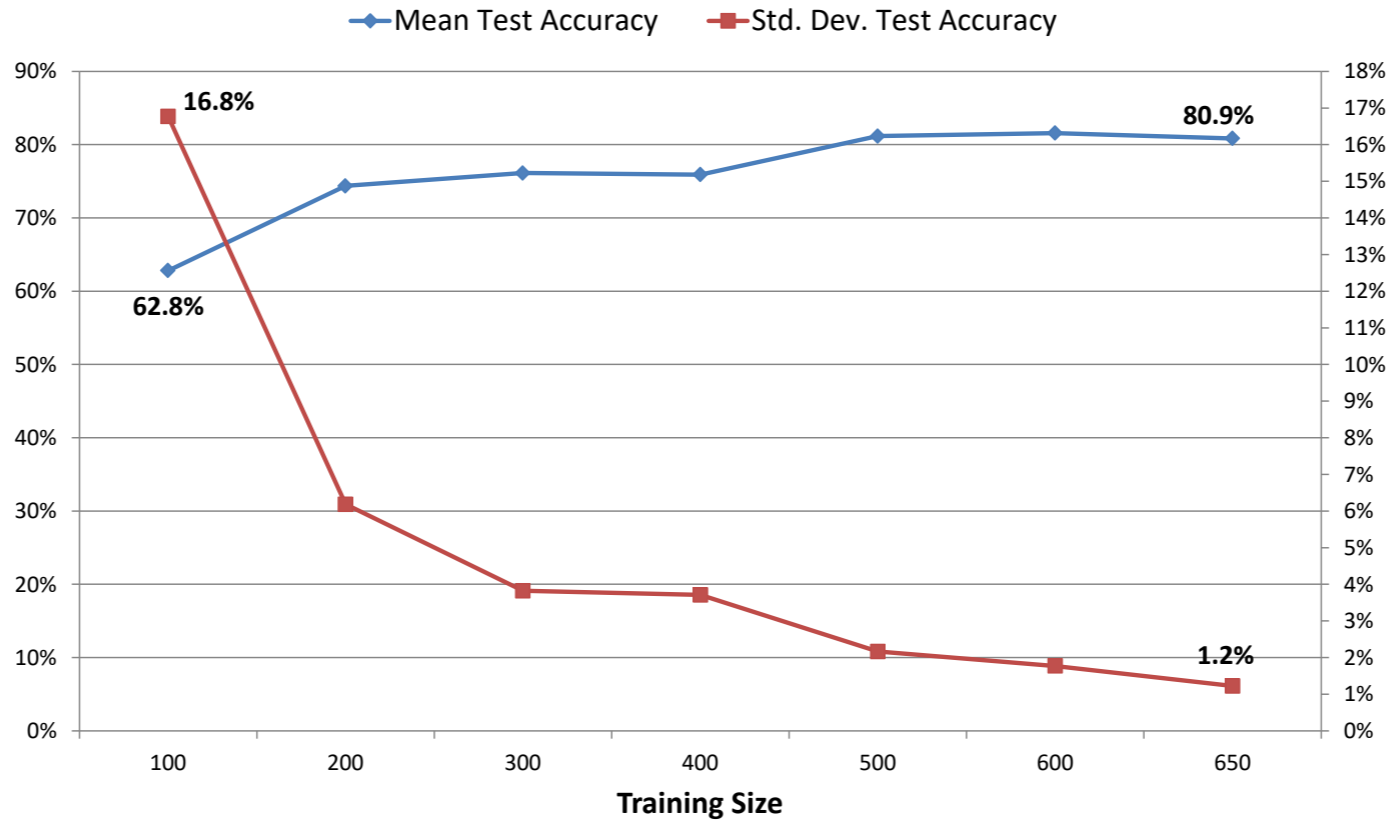
## *As Pelosi Takes Over, an Attempt to Revive the ‘Lost Art’ of Legislating*

She has agreed to a more open process, but Democrats who pushed for it may come to regret it.

# Experiment 2: HyperPartisan News Application

Naïve baseline model: CNN with BatchNorm and Dropout WITHOUT pretrained Glove

30 Trials each

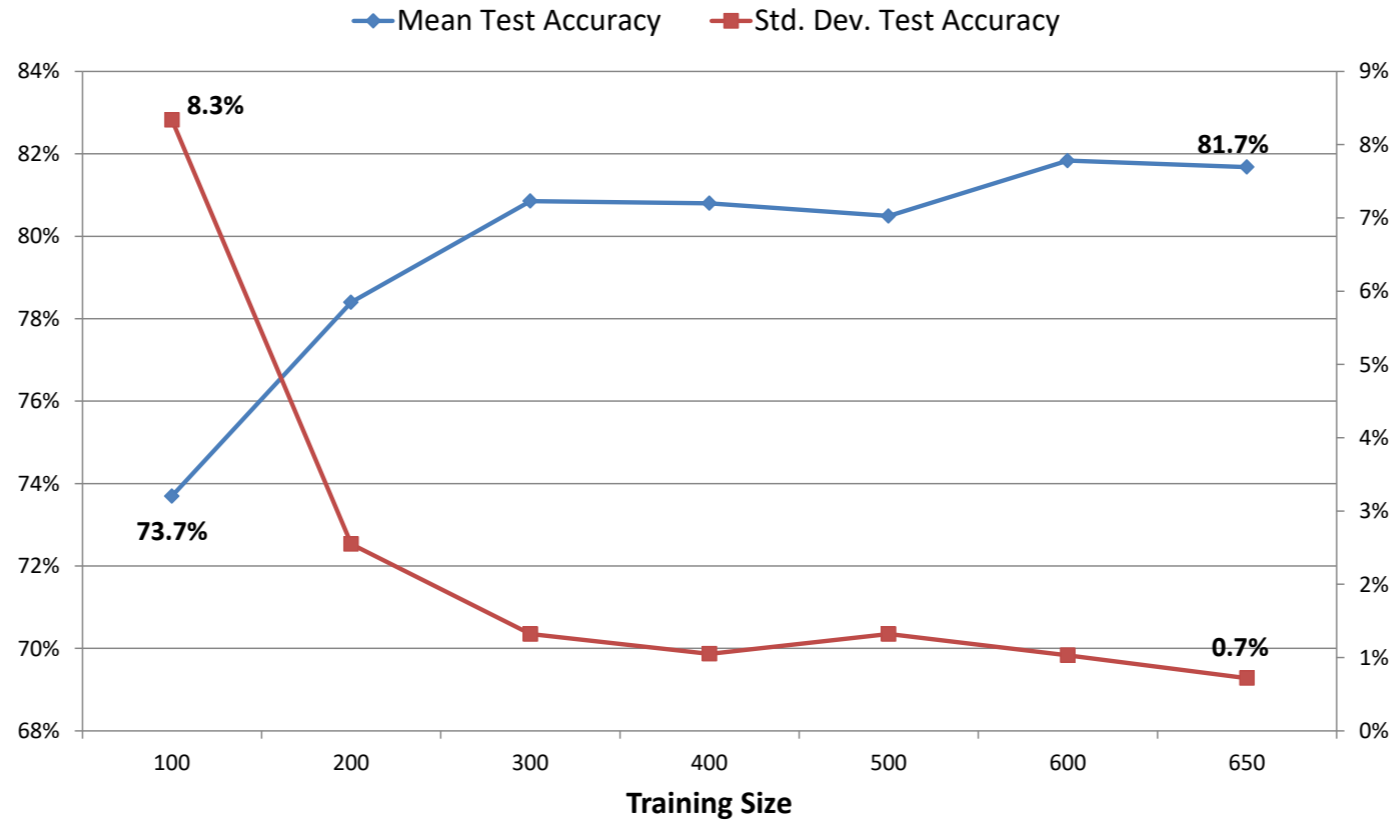


Source: UBS Evidence Lab

# Experiment 2: HyperPartisan News Application

More realistic baseline model: CNN with BatchNorm and Dropout WITH pretrained Glove

30 Trials each

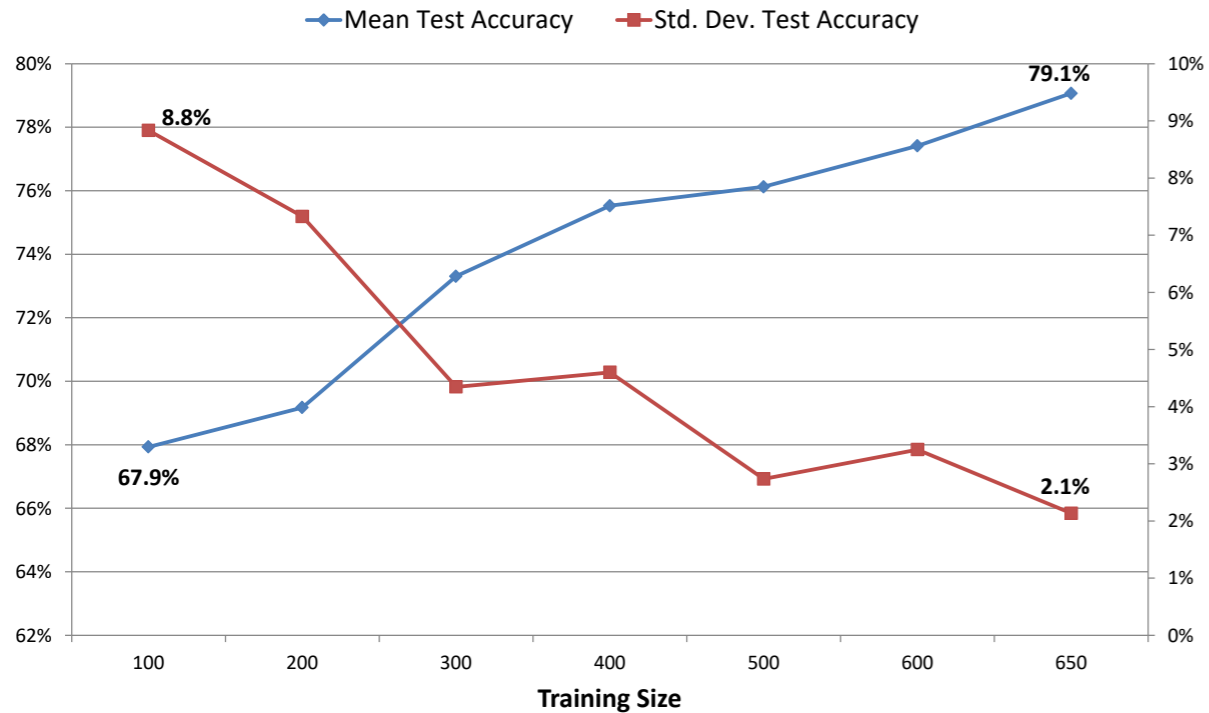


Source: UBS Evidence Lab

# Experiment 2: HyperPartisan News Application

## Universal Sentence Encoder: DAN

### Fine Tuning based Training – 30 Trials each



### Feature based Training – 30 Trials each



Source: UBS Evidence Lab

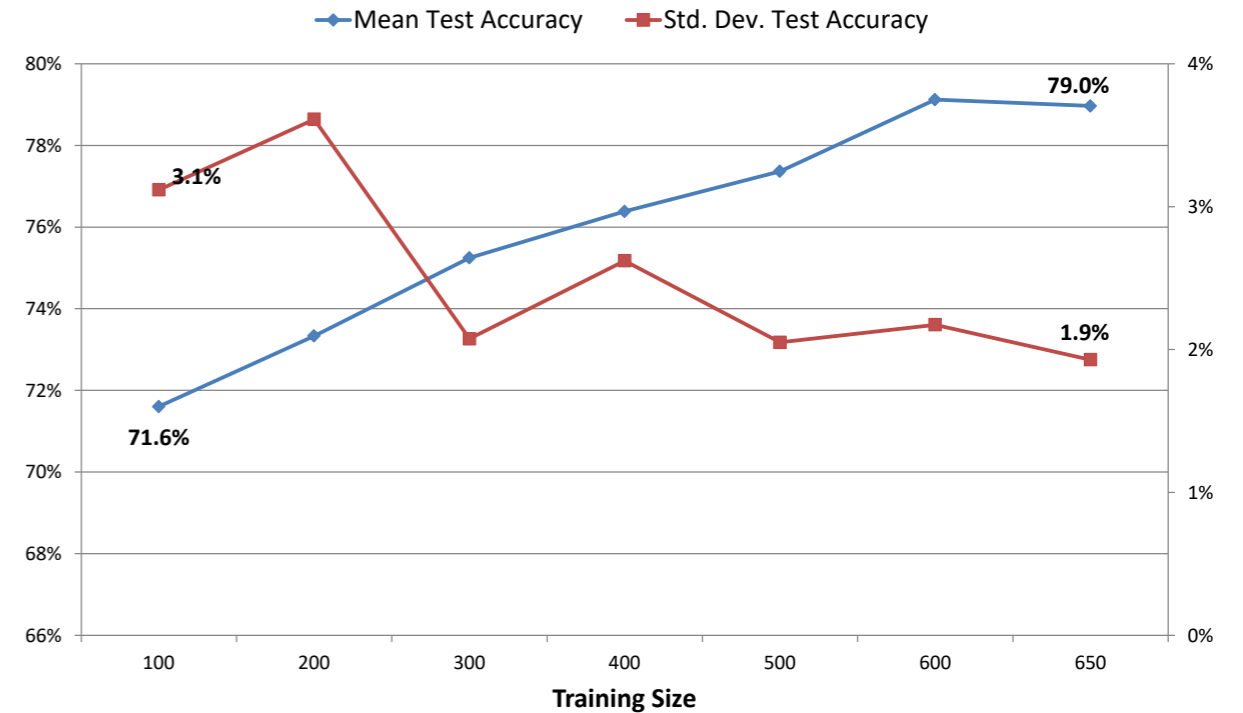
# Experiment 2: HyperPartisan News Application

ELMo

Fine Tuning based Training – 30 Trials each



Feature based Training – 30 Trials each



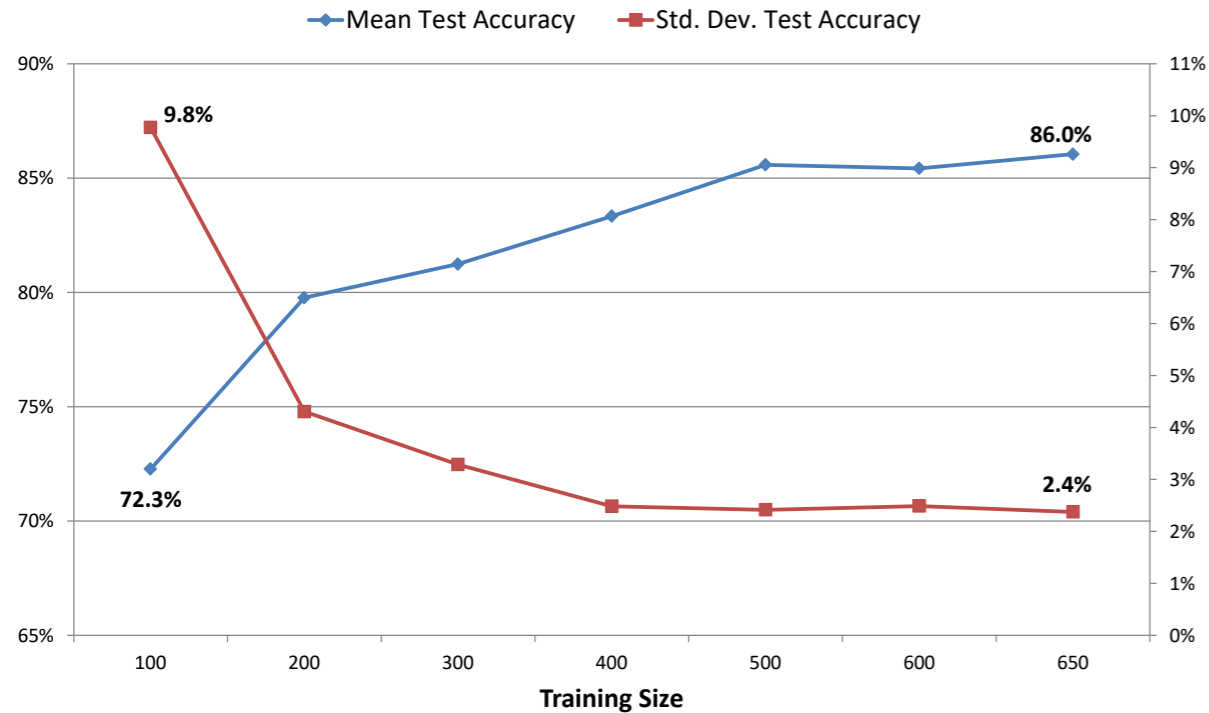
Source: UBS Evidence Lab



# Experiment 2: HyperPartisan News Application

BERT

Fine Tuning based Training – 30 Trials each



Feature based Training – 30 Trials each



Source: UBS Evidence Lab

# Experiment 2: HyperPartisan News Application

## Summary of Experimental Results

Model	100	200	300	400	500	600	650
Naïve Baseline	54%	70%	73%	73%	79%	80%	80%
Realistic Baseline	68%	76%	80%	80%	79%	81%	81%
USE - FT	62%	64%	70%	72%	74%	75%	77%
USE - FB	64%	68%	70%	71%	72%	72%	73%
ELMO - FT	66%	70%	68%	71%	73%	74%	74%
ELMO - FB	69%	71%	74%	74%	76%	77%	77%
BERT - FT	66%	76%	79%	81%	84%	83%	84%
BERT - FB	54%	69%	73%	75%	75%	77%	77%

$$\text{Adjusted Accuracy} = \frac{\text{Accuracy}}{(1 + \text{Stddev})}$$

# Results Summary

---

- There is no clear winner between finetune mode and feature mode
- BERT, in finetuning mode, is the best transfer learning model for big and small training sizes
- Feature mode for BERT however is much worse, especially for low training sizes.
- BERT in finetune mode beats a CNN model on entire training set size:
  - 87.1% vs 92.5% for IMDB (current SOTA is 95.4% with ULMFit)
  - 81% vs 86% for News



# Conclusions

---

- Bad News:
  - No clear winner between finetune mode and feature mode
  - Not all transfer learning architectures provide a clear advantage over CNN + Glove \*
- Good News:
  - BERT with finetuning works well for transfer learning model for low data problems
  - Achieved 50x sample efficiency for IMDB versus Naïve baseline
  - Achieved 3x sample efficiency for News versus Naïve baseline
  - With a training set of 100-150 samples per label using BERT, we could achieve near equal accuracy to baseline model using all available data
  - BERT achieves about 5-6% higher accuracy than baseline with all training data
  - Unsupervised language modeling on large datasets is a highly competitive method for pre-training

\*Robust hyperparameter tuning might make some improvements

# Future Work

---

- Apply concepts from ULMFit to BERT training
- More directed data selection procedures for incremental labeling
- Predicting when we have enough to the point of diminishing returns (on cost/benefit scale)
- How to make transfer learning work in the few-shot or zero-shot case

# Starter Code/Pre-trained Model Sources

---

- Baseline CNN + Glove: <https://github.com/tensorflow/models/tree/master/research>,  
<https://nlp.stanford.edu/projects/glove/>
- ELMo, USE models: Tensorflow Hub → <https://www.tensorflow.org/hub>
- BERT: <https://github.com/huggingface/pytorch-pretrained-BERT>

# Q & A

---

- Raghav Madhavan: [raghav.madhavan@ubs.com](mailto:raghav.madhavan@ubs.com)
- Hanoz Bhatena: [hanoz.bhatena@ubs.com](mailto:hanoz.bhatena@ubs.com)

# Appendix

---

Page intentionally left blank

# Sequence Autoencoders & LM Pre-training

---

- Recurrent Language Model:
  - Train a language model to predict the next word in a sequence using an LSTM/GRU cell
  - Given this trained model we can now use it on a downstream task like text classification
- Sequence autoencoder:
  - Train an LSTM encoder to embed a sentence into a single vector from which a second LSTM decoder can re-generate the input sentence.

# Universal Sentence Encoder – Tensorflow Hub Example

---

```
import tensorflow as tf
import tensorflow_hub as hub

embed = hub.Module("https://tfhub.dev/google/universal-sentence-encoder-large/3")
sentences = ["I like my phone", "Will it snow tomorrow?"]
with tf.Session() as session:
    session.run([tf.global_variables_initializer(), tf.tables_initializer()])
    sentences_embeddings = session.run(embed(sentences))
```

# BERT

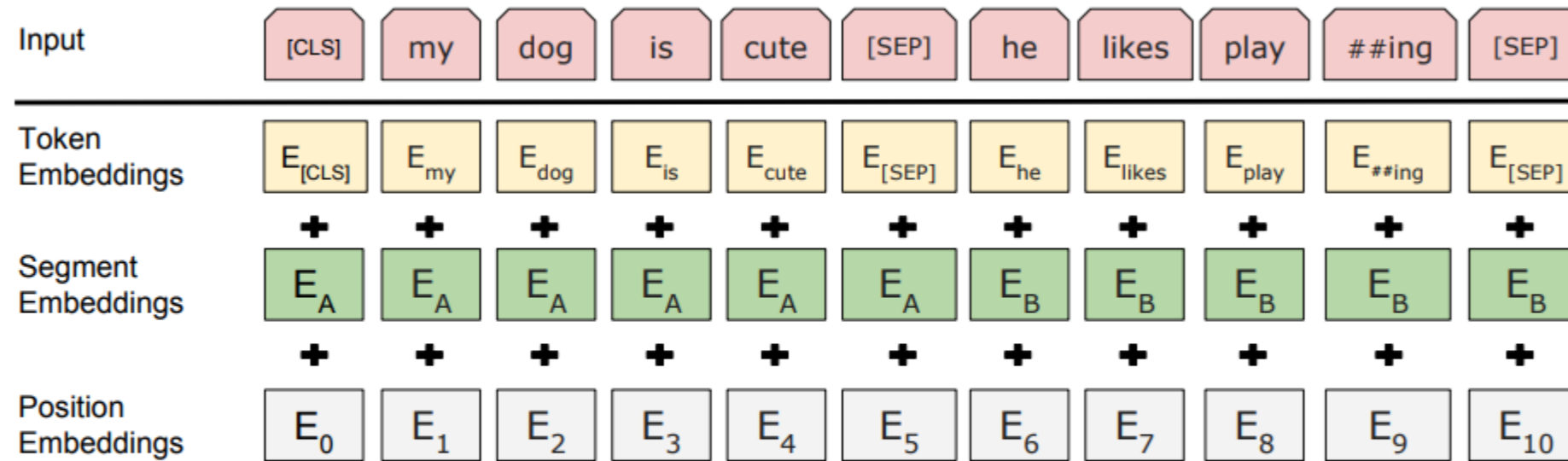


Figure 2: BERT input representation. The input embeddings is the sum of the token embeddings, the segmentation embeddings and the position embeddings.

Source: Original BERT paper



# BERT: Masked LM details

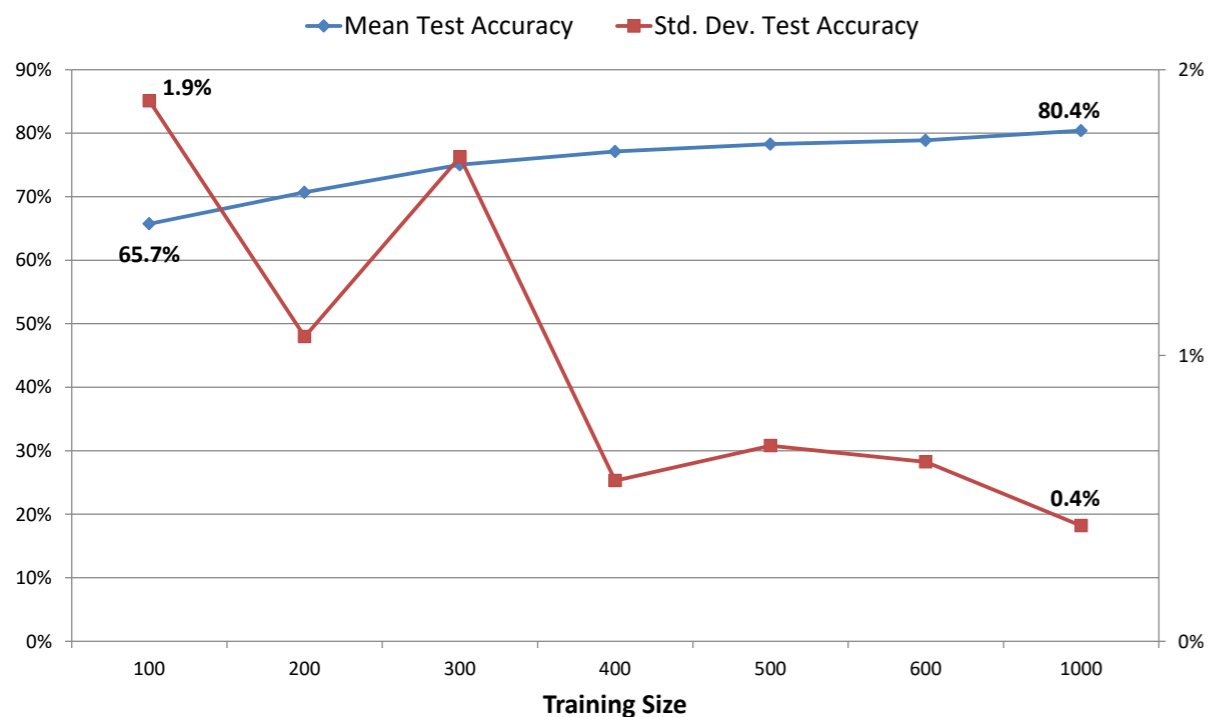
---

- One of the main innovative contributions is the bi-directional language model training using masking
- Typically when we use the term bi-directional, we are actually running two independent language models and concatenating hidden states
- However, BERT is able to achieve a truly bidirectional language model training by use of masking
- Replace a word/token with the [MASK] symbol and try to make the model learn to predict the token that should have been in the masked tokens's position
- 15% of tokens are chosen to be masked
- During training:
  - 80% of time replace word with [MASK] token
  - 10% of the time replace word with a random word
  - 10% of the time keep word unchanged so as to bias the representation to the real observed word

# Experiment 1: IMDB Rating Application

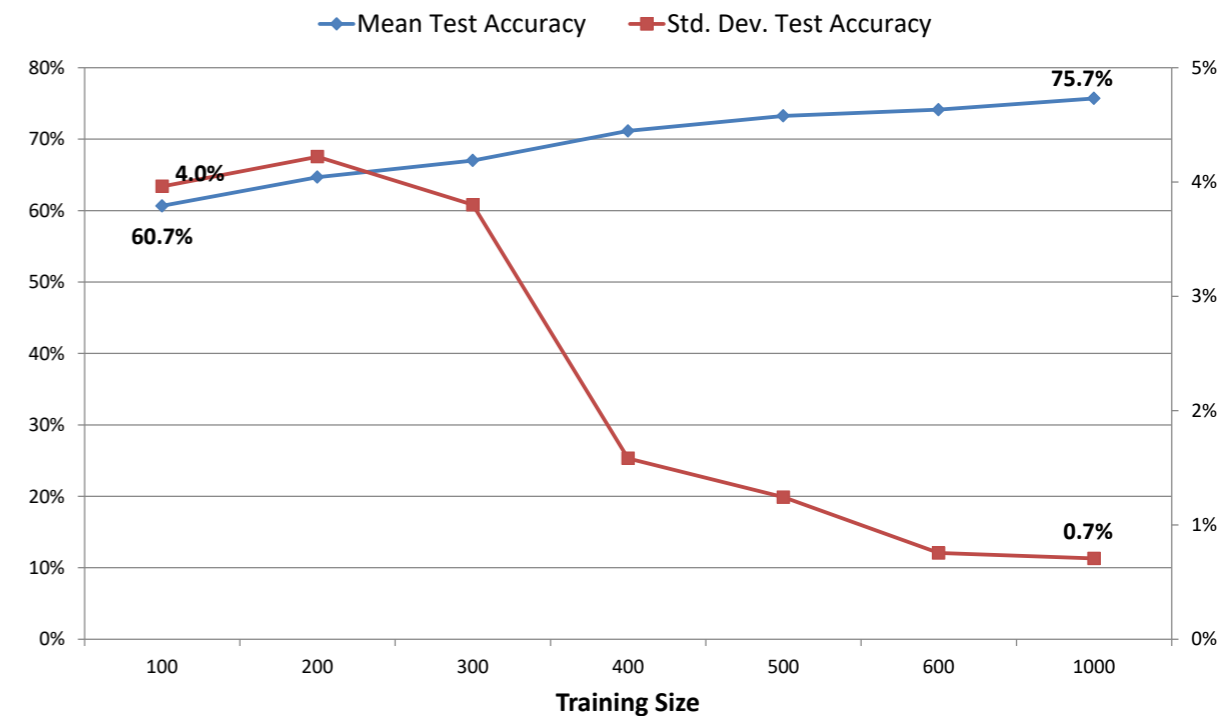
## N-gram Neural Network Language Model: NNLM

### Fine Tuning based Training – 10 Trials each



Using 25,000 training sample yields: 86.4%

### Feature based Training – 10 Trials each



Using 25,000 training sample yields: 79.1%

Source: UBS Evidence Lab

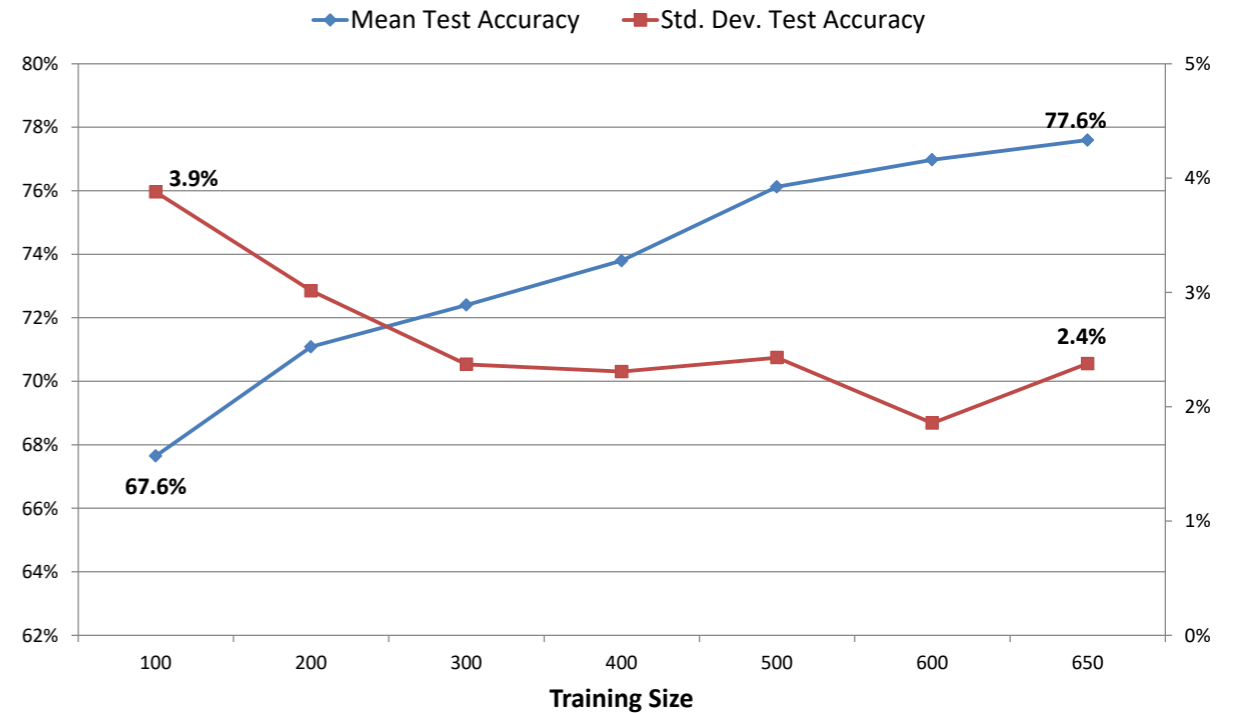
# Experiment 2: HyperPartisan News Application

## N-gram Neural Network Language Model: NNLM

### Fine Tuning based Training – 30 Trials each



### Feature based Training – 30 Trials each



Source: UBS Evidence Lab