Not all Textual Instances are Alike: Efficient NLP by Better Understanding of our Data

Roy Schwartz
Hebrew University of Jerusalem
SustainNLP 2021
Premise: Big Models
10,000X in 3 Years
Large Datasets
100,000X in 3 Years

![Graph showing the growth of large datasets over time]

- Supervised learning
- ELMo
- open-GPT2
- T5
Efficiency
Current Approaches
Efficiency
Current Approaches
Efficiency

Current Approaches

- **Model distillation**
  - Hinton et al. (2015); MobileBERT (Sun et al., 2019); DistilBERT (Sanh et al., 2019)

- **Pruning / Structural Pruning**
  - Han et al. (2016); SNIP (Lee et al., 2019); LTH (Frankle & Corbin, 2019); MorphNet (Gordon et al., 2018); Michel et al. (2019); LayerDrop (Fan et al., 2020); Dodge, Schwartz et al. (2019)

- **Quantization**
  - Gong et al. (2014); Q8BERT (Zafrir et al., 2019); Q-BERT (Shen et al., 2019)
Data in NLP

Basic Assumption: Instances are IID
Not all Instances are Alike

1. The movie was awesome.
2. I could definitely see why this movie received such great critiques, but at the same time I can’t help but wonder whether the plot was written by a 12 year-old or by an award-winning writer.
1. The movie was awesome.

2. I could definitely see why this movie received such great critiques, but at the same time I can’t help but wonder whether the plot was written by a 12 year-old or by an award-winning writer.
Outline

Not all Instances are Alike

• Efficient inference
  • Schwartz et al., ACL 2020

• Efficient training
  • Swayamdipta et al., EMNLP 2020

• Better masked language modeling for vision and language
  • Bitton et al., Findings of EMNLP 2021
Case Study 1: Efficient Inference
Schwartz et al., ACL 2020

Some instances require less processing than others
High-Level Idea

![Graph showing the relationship between Inference time and Accuracy. The graph has two axes: the x-axis represents Inference time ranging from 20 to 100, and the y-axis represents Accuracy ranging from 80 to 100. There are three data points: one at (20, 90), one at (60, 95), and one at (100, 100).]
High-Level Idea

![Graph showing the relationship between inference time and accuracy. The x-axis represents inference time, ranging from 20 to 100, and the y-axis represents accuracy, ranging from 80 to 100. There are three data points: one at (20, 80), one at (60, 90), and one at (100, 100).]
High-Level Idea

![Inference time vs. Accuracy graph](image-url)
High-Level Idea

Run an efficient model on “easy” instances,
and an expensive model on “hard” instances
High-Level Idea

Run an **efficient** model on “easy” instances, and an **expensive** model on “hard” instances.
Our Approach: Training Time

Layer n

Layer n-1

Layer n-2

Layer 2

Layer 1

Layer 0

Input

Prediction
Our Approach: Training Time

- Input
- Layer 0
- Layer 1
- Layer 2
- Layer n-2
- Layer n-1
- Layer n

Prediction
Our Approach: Test Time

Our approach involves testing the model layer by layer. The process starts with the input layer and progresses through each subsequent layer. At each layer, the model makes a prediction and checks for confidence. If the model is confident in its prediction, it early exits to save computational resources. If not, it moves to the next layer and repeats the process.
Our Approach: Test Time

Layer 0
Input

Layer 1

Layer 2

Layer n/2

Layer 1

Layer n

Prediction

Is confident?

Yes
Early exit

No

Prediction

Is confident?

Yes
Early exit

No

Prediction

Is confident?

Yes
Early exit

No
Our Approach: Test Time

Layer 0

Layer 1

Layer 2

Layer n/2

Layer I

Layer n

Input

Prediction

Is confident?

Yes → Early exit

No

Yes → Early exit

Prediction

Is confident?

Yes → Early exit

No

Prediction

Is confident?

Yes → Early exit

No
Our Approach: Test Time

- Input
  - Layer 0
  - Layer 1
  - Layer 2
  - Layer n/2
  - Layer n

Prediction:
- Is confident?
  - Yes: Early exit
  - No: Prediction (recursively)

- Prediction (recursively)
Our Approach: Test Time

Layer 0

Layer 1

Layer n/2

Layer n

Input

Layer 2

Layer 1

Layer n/2

Layer n

Prediction

Is confident?

Yes

Early exit

No

Prediction

Is confident?

Yes

Early exit

No

Prediction

Is confident?

Yes

Early exit

No

Prediction

Is confident?

Yes

Early exit

No

Prediction

Is confident?
Our Approach: Test Time

- **Input**
- **Layer 0**
- **Layer 1**
- **Layer 2**
- **Layer n/2**
- **Layer l**
- **Layer n**
- **Prediction**
  - Is confident?
    - Yes → Early exit
    - No
      - Is confident?
        - Yes → Early exit
        - No
          - Is confident?
            - Yes → Early exit
            - No
Calibrated Confidence Scores

- Interpret the calibrated softmax label scores as model confidence
  - We use temperature calibration (Guo et al., 2017)
Calibrated Confidence Scores

• Interpret the calibrated softmax label scores as model confidence
  • We use temperature calibration (Guo et al., 2017)

• Speed/accuracy tradeoff controlled by a single early-exit confidence threshold
Baselines

Standard baseline

Layer n

Layer n-1

Layer n-2

Layer 2

Layer 1

Layer 0

Input

Prediction
Baselines

Standard baseline

Layer n
Layer n-1
Layer n-2

Layer 2
Layer 1
Layer 0

Input

Prediction

Efficient baselines

Layer n
Layer n-1
Layer n-2

Layer 2
Layer 1
Layer 0

Input

Prediction

Layer n
Layer n-1
Layer n-2

Layer 2
Layer 1
Layer 0

Input

Prediction
Experimental Results:

Strong Baselines!
Experimental Results:

Strong Baselines!

![Graph showing accuracy and processing time]
Experimental Results:

Strong Baselines!

3 times faster, within 1% of full model
Better Speed/Accuracy Tradeoff

![Graph showing the tradeoff between processing time and accuracy. The graph illustrates how our model (blue line) compares to standard (red diamond) and efficiency (green circles) methods. The x-axis represents the processing time as a percentage of standard, and the y-axis represents accuracy as a percentage of standard. The graph shows a clear improvement in accuracy for our model compared to standard methods.]
Better Speed/Accuracy Tradeoff

![Graph showing the tradeoff between accuracy and processing time. The y-axis represents accuracy (% of std.), and the x-axis represents processing time (% of std.). There are two sets of data points: one for the standard method (red diamonds) and another for an efficient method (green circles). The graph illustrates how increasing processing time leads to an improvement in accuracy.]
Better Speed/Accuracy Tradeoff

Acc. (% of std., $\Rightarrow$)

Processing time (% of std., $\Leftarrow$)

- std.
- eff.
- our model
Better Speed/Accuracy Tradeoff

5 times faster, within 1% of full model
Better Speed/Accuracy Tradeoff

Similar results on SST, IMDB

5 times faster, within 1% of full model
More about our Approach

- No effective growth in parameters
  - < 0.005% additional parameters
More about our Approach

- No effective growth in parameters
  - < 0.005% additional parameters
- Training is not slower
More about our Approach

• No effective growth in parameters
  • < 0.005% additional parameters

• Training is not slower

• A single trained model provides multiple options along the speed/accuracy tradeoff
  • A single parameter: confidence threshold
More about our Approach

- No effective growth in parameters
  - $< 0.005\%$ additional parameters
- Training is not slower
- A **single** trained model provides multiple options along the speed/accuracy tradeoff
  - A single parameter: confidence threshold
- Caveat: requires batch size=1 during inference
Case Study 2: Efficient Training
Swayamdipta, Schwartz et al., EMNLP 2020

Some instances are more valuable for training than others
High-Level Idea

• *Divide the instances in a dataset into different groups*
• *Identify the groups that are *most valuable* for learning*
• *Train on those groups only, leading to *substantially faster training***
Training Dynamics

• Assume a model trained for K epochs
Training Dynamics

• Assume a model trained for $K$ epochs

• At each epoch, the model makes predictions on each training sample
  • This leads to a vector of size $K$ for each training instance
Training Dynamics

- Assume a model trained for K epochs
- At each epoch, the model makes predictions on each training sample
  - This leads to a vector of size K for each training instance
- We compute two measures on each vector:
  - Mean
  - Variability
Training Dynamics

Toy Example
Dataset Map Example
SNLI, RoBERTa-Large
Sample-Efficient Training

- *easy-to-learn* instances provide little value to training
- Can we use training dynamics to select the *most valuable* instances?
# Experiments

**WinoGrande, RoBERTa-Large**

<table>
<thead>
<tr>
<th></th>
<th>WINOG. Val. (ID)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100% train</td>
<td>79.7 ± 0.2</td>
</tr>
<tr>
<td>random</td>
<td>73.3 ± 1.3</td>
</tr>
<tr>
<td>forgetting</td>
<td>75.5 ± 1.3</td>
</tr>
<tr>
<td>AL-uncertainty</td>
<td>75.7 ± 0.8</td>
</tr>
<tr>
<td>AL-greedyK</td>
<td>74.2 ± 0.4</td>
</tr>
<tr>
<td>AFLite</td>
<td>76.8 ± 0.8</td>
</tr>
<tr>
<td><em>ambiguous</em></td>
<td><strong>78.7 ± 0.4</strong></td>
</tr>
</tbody>
</table>
# Experiments

**WinoGrande, RoBERTa-Large**

<table>
<thead>
<tr>
<th>Method</th>
<th>WINOG. Val. (ID)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100% train</td>
<td>79.7 ± 0.2</td>
</tr>
<tr>
<td>random</td>
<td>73.3 ± 1.3</td>
</tr>
<tr>
<td>forgetting</td>
<td>75.5 ± 1.3</td>
</tr>
<tr>
<td>AL-uncertainty</td>
<td>75.7 ± 0.8</td>
</tr>
<tr>
<td>AL-greedyK</td>
<td>74.2 ± 0.4</td>
</tr>
<tr>
<td>AFLite</td>
<td>76.8 ± 0.8</td>
</tr>
<tr>
<td>ambiguous</td>
<td>78.7 ± 0.4</td>
</tr>
</tbody>
</table>
### Experiments

**WinoGrande, RoBERTa-Large**

<table>
<thead>
<tr>
<th>WINOG. Val. (ID)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>100% train</td>
<td>79.7 ± 0.2</td>
</tr>
<tr>
<td>random</td>
<td>73.3 ± 1.3</td>
</tr>
<tr>
<td>forgetting</td>
<td>75.5 ± 1.3</td>
</tr>
<tr>
<td>AL-uncertainty</td>
<td>75.7 ± 0.8</td>
</tr>
<tr>
<td>AL-greedyK</td>
<td>74.2 ± 0.4</td>
</tr>
<tr>
<td>AFLite</td>
<td>76.8 ± 0.8</td>
</tr>
<tr>
<td>ambiguous</td>
<td>78.7 ± 0.4</td>
</tr>
</tbody>
</table>

33%
## Experiments

### WinoGrande, RoBERTa-Large

<table>
<thead>
<tr>
<th></th>
<th>WINOG. Val. (ID)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100% train</td>
<td>79.7 ± 0.2</td>
</tr>
<tr>
<td>random</td>
<td>73.3 ± 1.3</td>
</tr>
<tr>
<td>forgetting</td>
<td>75.5 ± 1.3</td>
</tr>
<tr>
<td>AL-uncertainty</td>
<td>75.7 ± 0.8</td>
</tr>
<tr>
<td>AL-greedyK</td>
<td>74.2 ± 0.4</td>
</tr>
<tr>
<td>AFLite</td>
<td>76.8 ± 0.8</td>
</tr>
<tr>
<td>ambiguous</td>
<td>78.7 ± 0.4</td>
</tr>
</tbody>
</table>

33%
# Experiments

## WinoGrande, RoBERTa-Large

<table>
<thead>
<tr>
<th></th>
<th>WINO G. Val. (ID)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100% train</td>
<td>79.7 ± 0.2</td>
</tr>
<tr>
<td>random</td>
<td>73.3 ± 1.3</td>
</tr>
<tr>
<td>forgetting</td>
<td>75.5 ± 1.3</td>
</tr>
<tr>
<td>AL-uncertainty</td>
<td>75.7 ± 0.8</td>
</tr>
<tr>
<td>AL-greedyK</td>
<td>74.2 ± 0.4</td>
</tr>
<tr>
<td>AFLite</td>
<td>76.8 ± 0.8</td>
</tr>
<tr>
<td><strong>ambiguous</strong></td>
<td><strong>78.7 ± 0.4</strong></td>
</tr>
</tbody>
</table>

33%
# Experiments

## WinoGrande, RoBERTa-Large

<table>
<thead>
<tr>
<th>WINOG. Val. (ID)</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>100% train</td>
<td>79.7 ± 0.2</td>
</tr>
<tr>
<td>random</td>
<td>73.3 ± 1.3</td>
</tr>
<tr>
<td>forgetting</td>
<td>75.5 ± 1.3</td>
</tr>
<tr>
<td>AL-uncertainty</td>
<td>75.7 ± 0.8</td>
</tr>
<tr>
<td>AL-greedyK</td>
<td>74.2 ± 0.4</td>
</tr>
<tr>
<td>AFLite</td>
<td>76.8 ± 0.8</td>
</tr>
<tr>
<td><strong>ambiguous</strong></td>
<td><strong>78.7 ± 0.4</strong></td>
</tr>
</tbody>
</table>

33%
## Experiments

**WinoGrande, RoBERTa-Large**

<table>
<thead>
<tr>
<th>Method</th>
<th>WINOg. Val. (ID)</th>
<th>WSC (OOD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100% train</td>
<td>79.7 ± 0.2</td>
<td>86.0 ± 0.1</td>
</tr>
<tr>
<td>Random</td>
<td>73.3 ± 1.3</td>
<td>85.6 ± 0.4</td>
</tr>
<tr>
<td>Forgetting</td>
<td>75.5 ± 1.3</td>
<td>84.8 ± 0.7</td>
</tr>
<tr>
<td>AL-uncertainty</td>
<td>75.7 ± 0.8</td>
<td>85.7 ± 0.8</td>
</tr>
<tr>
<td>AL-greedyK</td>
<td>74.2 ± 0.4</td>
<td>86.5 ± 0.5</td>
</tr>
<tr>
<td>AFLite</td>
<td>76.8 ± 0.8</td>
<td>86.6 ± 0.6</td>
</tr>
<tr>
<td>Ambiguous</td>
<td>78.7 ± 0.4</td>
<td>87.6 ± 0.6</td>
</tr>
</tbody>
</table>
## Experiments

**WinoGrande, RoBERTa-Large**

<table>
<thead>
<tr>
<th>Method</th>
<th>WINO G. Val. (ID)</th>
<th>WSC (OOD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100% train</td>
<td>79.7 ± 0.2</td>
<td>86.0 ± 0.1</td>
</tr>
<tr>
<td>random</td>
<td>73.3 ± 1.3</td>
<td>85.6 ± 0.4</td>
</tr>
<tr>
<td>forgetting</td>
<td>75.5 ± 1.3</td>
<td>84.8 ± 0.7</td>
</tr>
<tr>
<td>AL-uncertainty</td>
<td>75.7 ± 0.8</td>
<td>85.7 ± 0.8</td>
</tr>
<tr>
<td>AL-greedyK</td>
<td>74.2 ± 0.4</td>
<td>86.5 ± 0.5</td>
</tr>
<tr>
<td>AFLite</td>
<td>76.8 ± 0.8</td>
<td>86.6 ± 0.6</td>
</tr>
<tr>
<td>ambiguous</td>
<td>78.7 ± 0.4</td>
<td>87.6 ± 0.6</td>
</tr>
</tbody>
</table>
Experiments

WinoGrande, RoBERTa-Large

<table>
<thead>
<tr>
<th></th>
<th>WINOG. Val. (ID)</th>
<th>WSC (OOD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100% train</td>
<td>79.70.2</td>
<td>86.00.1</td>
</tr>
<tr>
<td>random</td>
<td>73.31.3</td>
<td>85.60.4</td>
</tr>
<tr>
<td>forgetting</td>
<td>75.51.3</td>
<td>84.80.7</td>
</tr>
<tr>
<td>AL-uncertainty</td>
<td>75.70.8</td>
<td>85.70.8</td>
</tr>
<tr>
<td>AL-greedyK</td>
<td>74.20.4</td>
<td>86.50.5</td>
</tr>
<tr>
<td>AFLite</td>
<td>76.80.8</td>
<td>86.60.6</td>
</tr>
<tr>
<td>ambiguous</td>
<td>78.70.4</td>
<td>87.60.6</td>
</tr>
</tbody>
</table>

33% Similar results on SNLI, MNLI, QNLI
Recap

• Some instances contribute more to learning

• We select the ones with the highest variance in confidence level across training

• 3x reduction in training time
  • Minimal reduction in ID performance
  • Improvement on OOD performance

• Limitations
  • Model-dependent
  • Requires training on the full dataset first
Case Study 3:
Efficient Pre-training for Vision and Language
Bitton, Stanovsky, Elhadad & Schwartz, Findings of EMNLP 2021

Some words are more valuable for pre-training than others
MLM in Vision and Language

- Virtually all vision pre-training works (Shin et al., 2021) follow BERT and randomly mask 15% of the tokens
MLM in Vision and Language

• Virtually all vision pre-training works (Shin et al., 2021) follow BERT and randomly mask 15% of the tokens.

• Of the masked tokens, roughly one half are stop-words or punctuation.
MLM in Vision and Language

• Virtually all vision pre-training works (Shin et al., 2021) follow BERT and randomly mask 15% of the tokens

• Of the masked tokens, roughly one half are stop-words or punctuation

• We propose better masking strategies for V&L MLM
MLM in Vision and Language

- Virtually all vision pre-training works (Shin et al., 2021) follow BERT and randomly mask 15% of the tokens.
- Of the masked tokens, roughly one half are stop-words or punctuation.
- We propose better masking strategies for V&L MLM.
- See Yonatan’s talk for more details!
Improved Downstream Performance
Especially on Low-Resource Settings
Improved Downstream Performance
Especially on Low-Resource Settings

![Graph showing VQA performance vs. training data percentage. The graph indicates improved performance with different masking strategies.]
Improved Downstream Performance
Especially on Low-Resource Settings

![Graph showing accuracy vs % of training data for different masking strategies.](image)
Improved Downstream Performance
Especially on Low-Resource Settings

Similar accuracy, twice as fast
Improved Downstream Performance
Especially on Low-Resource Settings

Similar results on GQA, NLVR2

Similar accuracy, twice as fast
Not all Instances are Alike

Recap

- Efficient **inference** by selecting *the right tool for the job*
- Efficient **fine-tuning** by selecting the most *ambiguous* examples
- Efficient multi-modal **pre-training** by better *masking strategies*
Amazing Collaborators!
Not all instances are alike

Recap

- Efficient **inference** by selecting *the right tool for the job*
- Efficient **fine-tuning** by selecting the most *ambiguous* examples
- Efficient multi-modal **pre-training** by better *masking strategies*