Natural Language Understanding, Pretrained Language Models

Karl Stratos

Rutgers University
Natural Language Understanding (NLU) Tasks

- Tasks that (1) cannot be solved by just using word-level patterns (must use logic, predicate/argument structure, etc.), (2) require “common sense” outside task-specific supervision

- Tasks not considered NLU
  - Topic classification: Bag-of-words linear classifier works fine
  - Short translation: Mapping self-contained, no need for much external knowledge

- Tasks considered NLU
  - Sentiment analysis: A few instances do require genuine language understanding
  - Natural language inference (NLI): “If Liz likes John, is it the case that Liz loves John?”
  - Question answering: “Why does Queen Elizabeth sign her name Elizabeth R?”
  - Coreference resolution: “The trophy doesn’t fit in the suitcase because [it]’s too big.”
Natural Language Inference (NLI)

- Can be framed as sentence-pair classification
  - **Input.** (Premise, Hypothesis)
  - **Output.** Entailment (E), contradiction (C), or neutral (N)

- Examples *(Bowman et al., 2015)*
  - \( (A \text{ soccer game with multiple males playing,} \text{ Some men are playing a sport.}) \to E \)
  - \( (A \text{ black race car starts up in front of a crowd of people,} \text{ A man is driving down a lonely road.}) \to C \)
  - \( (An \text{ older and younger man smiling,} \text{ Two men are smiling and laughing at the cats playing on the floor.}) \to N \)

- MNLI *(Williams et al., 2018)*: 393k training instances, 20k test
  - Annotation by crowdsourcing (relatively easy for humans)
  - Human accuracy: 92
  - CBOW-based classifier accuracy: 56
  - BiLSTM-based classifier accuracy: \( \approx 70 \)
Question Answering (QA)

- Can be framed as predicting an answer span in a passage given a passage-question pair
- Early work: SQuAD dataset (Rajpurkar et al., 2016)

Input.
- Passage: “In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravity. The main forms of precipitation include drizzle, rain, sleet, snow, graupel and hail . . .”
- Question: “What causes precipitation to fall?”

Output. Span (17, 18), corresponding to “gravity”

Many challenges in defining QA tasks
- Unlike MT, difficult to obtain natural data. SQuAD asked annotators to create questions answerable by passage
- Spurious occurrences of answer string (e.g., “three” can appear in irrelevant context)
Example: Natural Questions Dataset (Kwiatkowski et al., 2019)

Example 1
Question: what color was john wilkes booth’s hair
Wikipedia Page: John_Wilkes_Booth
Long answer: Some critics called Booth “the handsomest man in America” and a “natural genius”, and noted his having an “astonishing memory”; others were mixed in their estimation of his acting. He stood 5 feet 8 inches (1.73 m) tall, had jet-black hair, and was lean and athletic. Noted Civil War reporter George Alfred Townsend described him as a “muscular, perfect man” with “curling hair, like a Corinthian capital”.
Short answer: jet-black

Example 2
Question: can you make and receive calls in airplane mode
Wikipedia Page: Airplane_mode
Long answer: Airplane mode, aeroplane mode, flight mode, offline mode, or standalone mode is a setting available on many smartphones, portable computers, and other electronic devices that, when activated, suspends radio-frequency signal transmission by the device, thereby disabling Bluetooth, telephony, and Wi-Fi. GPS may or may not be disabled, because it does not involve transmitting radio waves.
Short answer: BOOLEAN:NO

Example 3
Question: why does queen elizabeth sign her name elizabeth r
Wikipedia Page: Royal_sign-manual
Long answer: The royal sign-manual usually consists of the sovereign’s regnal name (without number, if otherwise used), followed by the letter R for Rex (King) or Regina (Queen). Thus, the signs-manual of both Elizabeth I and Elizabeth II read Elizabeth R. When the British monarch was also Emperor or Empress of India, the sign manual ended with R I, for Rex Imperator or Regina Imperatrix (King-Emperor/Queen-Empress).
Short answer: NULL

- 307k training instances, 7.8k evaluation
- Input. (Question, Wikipedia Page)
- Questions: Real Google queries
- Output.
  1. Long answer: Either a paragraph that answers the question, or not-answerable
  2. Short answer: Either a short span (e.g., entity), yes/no, or null
- F1 evaluation (long answer, dev)
  - Human: 73.4
  - DocumentQA (BiRNN/attention, init with pretrained word embeddings): 46.1
Coreference Resolution (Coref)

- General coref: Identify and cluster mentions based on referenced entities
  
  We are looking for a region of central Italy bordering the Adriatic Sea. The area is mostly mountainous and includes Mt. Corno, the highest peak of the mountain range. It also includes many sheep and an Italian entrepreneur has an idea about how to make a little money of them.

- Simplification: Winograd Schema Challenge (WSC) (Levesque et al., 2011)
  
  - The drain is clogged with hair. It has to be cleaned.
  - The drain is clogged with hair. It has to be removed.

- Reduction to NLI (WNLI)
  
  \[
  \begin{align*}
  \text{The drain is clogged with hair, The hair has to be cleaned} & \rightarrow C \\
  \text{The drain is clogged with hair, The drain has to be cleaned} & \rightarrow E \\
  \text{The drain is clogged with hair, The hair has to be removed} & \rightarrow E \\
  \text{The drain is clogged with hair, The drain has to be removed} & \rightarrow C
  \end{align*}
  \]

- WNLI: 634 training instances, 146 test
  
  - Human accuracy 96
  - Any neural model trained from scratch: 65.1 (random)
Other NLU Tasks

- **Sentence similarity**: Formulated as sentence-pair regression
  - (A person is combing a cat hair., A person is brushing a cat.) $\rightarrow$ 4.4
  - (A man is cutting up a potato., A man is cutting up carrots.) $\rightarrow$ 2.4
  - (A boy is riding a horse., A monkey is riding a bus.) $\rightarrow$ 0.4

- **STS-B dataset**: 7k training examples, 1.4k test.
- **Performance (correlation)**: Human 92.7, BiLSTM 65-70

- **Linguistic acceptability**: Single-sentence binary classification (grammatical vs ungrammatical)
  - She voted for herself. $\rightarrow$ 1
  - Maryann should leaving. $\rightarrow$ 0
  - Kim persuaded it to rain. $\rightarrow$ 0
  - Books were sent to each other by the students. $\rightarrow$ 0

- **CoLA (Warstadt et al., 2018)**: 8.5k training examples, 1k test.
- **Performance (correlation)**: Human 66.4, BiLSTM 15
NLU Benchmarks

- Collection of tasks for testing NLU capabilities of a system
- Example: GLUE (Wang et al, 2018)

<table>
<thead>
<tr>
<th>Corpus</th>
<th>Train</th>
<th>Test</th>
<th>Task</th>
<th>Metrics</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Single-Sentence Tasks</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CoLA</td>
<td>8.5k</td>
<td>1k</td>
<td>acceptability</td>
<td>Matthews corr.</td>
<td>misc.</td>
</tr>
<tr>
<td>SST-2</td>
<td>67k</td>
<td>1.8k</td>
<td>sentiment</td>
<td>acc.</td>
<td>movie reviews</td>
</tr>
</tbody>
</table>

|                         | Similarity and Paraphrase Tasks |                  |                  |                 |
| MRPC                   | 3.7k                            | paraphrase       | acc./F1          | news            |
| STS-B                  | 7k                              | sentence similarity | Pearson/Spearman corr. | misc.         |
| QQP                    | 364k                            | paraphrase       | acc./F1          | social QA questions |

| Inference Tasks |                  |                  |                  |                 |
| MNLI            | 393k              | 20k              | NLI              | matched acc./mismatched acc. | misc. |
| QNLI            | 105k              | 5.4k             | QA/NLI          | acc.             | Wikipedia |
| RTE             | 2.5k              | 3k               | NLI              | acc.             | news, Wikipedia |
| WNLI            | 634               | 146              | coreference/NLI | acc.             | fiction books |

- Only train/dev data released: System submits predictions online to receive test performance
- Single score by macro-average
  - Human GLUE score: 87.1
  - BiLSTM GLUE score: 63.7
- Typically consider simple classification/regression tasks
  - Complex tasks like full-fledged QA and coref not included, must be considered additionally
Need for Transfer Learning

- NLU tasks, and other downstream tasks, supply limited supervision ($\approx 300k$ labeled examples at most)
- We can’t train a model from scratch for each task and expect it to develop general language understanding capabilities
- Solution: **Transfer Learning**
  - Use knowledge acquired to solve task $A$ to help better solve a related task $B$
- In particular: **Unsupervised** transfer learning
  - $A$ doesn’t need supervision
  - Form of semi-supervised learning (lots of unlabeled data, small labeled data)
- Central question: What task in NLP can we train a model for with no annotation, yet it’s closely related to many downstream tasks?
Pretrained Neural Language Models

- Family of neural LMs “pretrained” on a large quantity of unlabeled text
- Given a downstream task, copy the pretrained LM weights and “finetune” them on a small quantity of labeled data
  - Variations possible: Hold pretrained weights fixed, only train a new classification layer
- We don’t necessarily care about pretraining itself, as long as the resulting model is useful for downstream tasks.
  - Flexibility in designing the pretraining objective
- Some landmarks: Word2vec, ELMo, BERT, GPTs
  - Initial works like word2vec only considered pretraining word embeddings
  - Later works consider pretraining an entire LM capable of producing contextual word embeddings
Word2Vec (Mikolov et al., 2013): CBOW With Negative Sampling

- Given vocab $\mathcal{V}$ and dimension $d$, learn
  - Word embedding matrix: $W = [w_1 \ldots w_{|\mathcal{V}|}] \in \mathbb{R}^{d \times |\mathcal{V}|}$
  - Context word embedding matrix: $C' = [c_1 \ldots c_{|\mathcal{V}|}] \in \mathbb{R}^{d \times |\mathcal{V}|}$

- Training: Draw a random $n$-grams $(x_1 \ldots x_n)$ from a corpus with middle word $x_{\text{mid}}$. Set

\[
c_{\text{cbow}} = \frac{1}{n - 1} \sum_{i=1: i \neq \text{mid}}^{n} c_{x_i}
\]

Draw $K$ random words $x_{1}^{\text{neg}} \ldots x_{K}^{\text{neg}} \sim q$ where $q$ is some distribution over $\mathcal{V}$ (e.g., empirical unigram distribution). Take a gradient step on the single-example loss

\[
\hat{J}_{\text{single}}(W, C') = -\log \sigma(w_{x_{\text{mid}} \circ c_{\text{cbow}}}^\top) - \sum_{k=1}^{K} \log \sigma(-w_{x_{k}^{\text{neg}} \circ c_{\text{cbow}}}^\top)
\]
Distributional Word Representations

... this dog is a poodle...
...love my poodle...
...poodle and schnauzer...
...terrier is a dog...
...love your terrier...
...schnauzer, or terrier...

frog is an amphibian...
...frog from predators...
...cold-blooded, a frog...

"You shall know a word by the company it keeps." -Firth
Word2Vec in Practice

- Can be viewed as a stripped down LM
  - Predict what the middle word is given a bag of context words
  - Approximate cross-entropy loss by negative sampling (must specify number of negatives, e.g., \( K = 5 \))
- Efficient training, CPU friendly, parallelizable over corpus with asynchronous updates
  - Only a few hours to train on the entire Wikipedia corpus (3 billion tokens, vocab size > 100k)
- Once trained, use \( w_x \in \mathbb{R}^d \) as embedding of word \( x \in \mathcal{V} \)
  - Typically discard context embeddings \( c_x \)
- In a downstream task, initialize word embeddings with \( w_x \)
  - Significant improvement over randomly initialized word embeddings if labeled data is small
  - E.g., CoNLL 2003 NER performance using BiLSTM-CRF (Lample et al., 2016): \( 83.6 \rightarrow 90.9 \)
- Other non-neural word embedding techniques: GloVe (Pennington et al., 2014), spectral (Stratos et al., 2015)
  - All based on modeling distributional similarity between words by compressing context, have similar quality
Nearest Neighbor Examples

Top-8 words with highest cosine similarity using spectral word embeddings

<table>
<thead>
<tr>
<th>rochester</th>
<th>seattle</th>
<th>yahoo</th>
<th>starbucks</th>
<th>lol</th>
</tr>
</thead>
<tbody>
<tr>
<td>binghamton</td>
<td>tacoma</td>
<td>linkedin</td>
<td>dunkin</td>
<td>yeah</td>
</tr>
<tr>
<td>albany</td>
<td>portland</td>
<td>msn</td>
<td>mcdonalds</td>
<td>heh</td>
</tr>
<tr>
<td>hartford</td>
<td>washington</td>
<td>facebook</td>
<td>mcdonald’s</td>
<td>kidding</td>
</tr>
<tr>
<td>utica</td>
<td>denver</td>
<td>digg</td>
<td>domino’s</td>
<td>thats</td>
</tr>
<tr>
<td>syracuse</td>
<td>oakland</td>
<td>aol</td>
<td>applebee’s</td>
<td>damn</td>
</tr>
<tr>
<td>elmira</td>
<td>baltimore</td>
<td>google</td>
<td>7-eleven</td>
<td>ahh</td>
</tr>
<tr>
<td>bridgeport</td>
<td>chicago</td>
<td>friendster</td>
<td>kfc</td>
<td>gosh</td>
</tr>
<tr>
<td>newark</td>
<td>cleveland</td>
<td>orkut</td>
<td>walmart</td>
<td>kinda</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>smile</th>
<th>frown</th>
<th>1</th>
<th>1945</th>
<th>second</th>
</tr>
</thead>
<tbody>
<tr>
<td>smiles</td>
<td>frowns</td>
<td>2</td>
<td>1944</td>
<td>third</td>
</tr>
<tr>
<td>smiling</td>
<td>frowned</td>
<td>3</td>
<td>1943</td>
<td>fourth</td>
</tr>
<tr>
<td>grin</td>
<td>disapprove</td>
<td>4</td>
<td>1942</td>
<td>fifth</td>
</tr>
<tr>
<td>wide-eyed</td>
<td>cringe</td>
<td>5</td>
<td>1941</td>
<td>first</td>
</tr>
<tr>
<td>laugh</td>
<td>discourages</td>
<td>6</td>
<td>1946</td>
<td>sixth</td>
</tr>
<tr>
<td>cheerful</td>
<td>overreact</td>
<td>8</td>
<td>1940</td>
<td>seventh</td>
</tr>
<tr>
<td>eyes</td>
<td>detest</td>
<td>7</td>
<td>1939</td>
<td>eighth</td>
</tr>
<tr>
<td>grinning</td>
<td>forbid</td>
<td>9</td>
<td>1947</td>
<td>ninth</td>
</tr>
</tbody>
</table>
Limitations of Pretrained Word Embeddings

- **Non-contextual**: “saw” below gets the same word embedding
  
  the man saw the cut
  the saw cut the man

- Can use contextualizer on top like BiLSTMs in finetuning, but no transfer learning for that module

- Not helpful when downstream task has enough training data
  
  Example: MT, similar performance with random vs pretrained word embeddings

- Intuition: Generic word similarity is useful, but definitely not enough for general language understanding

- Natural next step: Transfer an entire LM, rather than just lookup tables
  
  Took a while before this happened because of “word embedding inertia”

- Much harder to transfer a contextual encoder
ELMo (Peters et al., 2017)

- **Embeddings from Language Models**
  - One of the first truly successful pretrained LMs for transfer learning, building on earlier works like
    - CoVe (McCann et al., 2017): Transfer learning by MT
    - TagLM (Peters et al., 2017): Transfer learning also by bidirectional LM. ELMo uses more layers and better techniques
- **Character-level input**
  - Run CNN over characters instead of having a static embedding for each word
  - Prediction is still word level
- **Backward LM that encodes context to the right, trained jointly**
- **Light-weight scheme to tailor ELMo embeddings for downstream tasks without finetuning ELMo itself**
Character-Level Input

- Each word $w$ treated as a sequence of characters $c \in C$ where $|C| = 262$ (UTF-8 encoding)
- Character embedding dimension 16: input matrix $C \in \mathbb{R}^{262 \times 16}$
- CNN filter sizes 1–7 with increasing filter numbers (32...1024) and max pooling: Outputs $u_w \in \mathbb{R}^{2048}$

- Final word rep: $v_w = \text{Feedforward}(\text{Highway}^2(u_w)) \in \mathbb{R}^{512}$
- This is the input to LSTMs, shared between forward/backward LMs

(Image credit: Petr Lorenc)
Bidirectional Language Modeling

▶ **Forward LM**
  - Two-layer LSTM cell: Input dim 512, cell state dim 4096 but hidden state dim projected back to 512

▶ **Backward LM**: Same architecture but distinct parameters

▶ Shared classification layer $W \in \mathbb{R}^{512 \times V}$ where $V = 793471$ vocab extracted from 1 Billion Word Benchmark dataset

▶ Loss: Sum of forward and backward LM losses

$$\hat{J}_x(\theta) = - \sum_{t=1}^{T} \left( \log p(x_t|x_{<t}; \theta_{\text{cnn}}, \theta_{\text{forward}}, \theta_{\text{softmax}}) + \log p(x_t|x_{>t}; \theta_{\text{cnn}}, \theta_{\text{backward}}, \theta_{\text{softmax}}) \right)$$
The ELMo Embeddings

- ELMo parameters frozen after pretraining
- Given sentence \(x_1 \ldots x_T\), running ELMo yields
  1. Word reps \(v_{x_1} \ldots v_{x_T} \in \mathbb{R}^{512}\) from CNN, can be precomputed
  2. Forward LSTM hidden states \(\vec{h}_1^{(l)} \ldots \vec{h}_T^{(l)} \in \mathbb{R}^{512}\) for each layer \(l = 1, 2\) where \(\vec{h}_t^{(l)}\) is a function of \(x_{\leq t}\)
  3. Backward LSTM hidden states \(\vec{h}_1^{(l)} \ldots \vec{h}_T^{(l)} \in \mathbb{R}^{512}\) for each layer \(l = 1, 2\) where \(\vec{h}_t^{(l)}\) is a function of \(x_{\geq t}\)
- 1024-dimensional contextual embedding of \(t\)-th word

\[
\text{ELMo}_t = \gamma \left( \alpha_0 \begin{bmatrix} v_{x_t} \\ v_{x_t} \end{bmatrix} + \alpha_1 \begin{bmatrix} \vec{h}_t^{(1)} \\ \vec{h}_t^{(1)} \end{bmatrix} + \alpha_2 \begin{bmatrix} \vec{h}_t^{(2)} \\ \vec{h}_t^{(2)} \end{bmatrix} \right)
\]

Introducing learnable scalar parameters \(\gamma, \alpha_l \in \mathbb{R}\) to scale embeddings from different layers for target task

- In a downstream task, just concatenate with initial word embedding

  - E.g., Input to RNN is word embeddings concat with \(\text{ELMo}_t\).
Results

- 10 epochs on 1B Word Benchmark (2 weeks on 3 GPUs)
  - 800 million tokens of news data, vocab size $800k$
- Append ELMo embeddings at input in various baseline models

(Image credit: Isha Salian)

CS 533: Natural Language Processing
Limitations of ELMo

- $\textbf{ELMo}_t \in \mathbb{R}^{1024}$ encodes both left/right context, but shallowly bidirectional

  (not bidirectional until later)

- Only transferring frozen contextual embeddings
  - Must train task-specific encoders like LSTMs on top

- How can we pretrain an LM that’s deeply bidirectional and almost “sufficient” on its own?
BERT (Devlin et al., 2019)

- Bidirectional Encoder Representations from Transformers
- Insight: Pretrain an LM in such a way that it’s “almost” the same as how it’ll be used for downstream tasks
- How do we use an NLP model for downstream tasks?
  1. Apply powerful transformation on tokens to get contextual token embeddings.
  2. Add a linear classifier on top.
- We want to pretrain an LM for 1, without limiting it to forward or backward token prediction
- Central question: How can we do language modeling while “seeing” the whole input text?
Masked Language Modeling (MLM)

- Mask out tokens (wordpieces) at random
  - the man went to the [MASK] to buy a [MASK] of milk
  - The model receives the input above and predict what the missing words are: “store”, “gallon”
  - Crucially, can use context to the right all the time!
- Need to be careful
  - Too little masking: too expensive to train
  - Too much masking: not enough context
  - Test time: no [MASK] input, so training should also handle no [MASK] input sometimes
- BERT masking scheme: Given input text,
  - Choose 15% of tokens uniformly at random
  - For each chosen token, replace it with [MASK] 80% of the time, a random token 10% of the time, and leave it unchanged 10% of the time.
Details of BERT

- Wordpiece tokenization: Vocab size $30k$ (cased/uncased versions)
- **Transformer encoder**
  - bert-base: 12 layers, 12 attention heads, 110m parameters
  - bert-large: 24 layers, 16 attention heads, 340m parameters
- Input: Sentence pair (marked at input by additive embeddings), predict consecutive (50% random) in addition to MLM
- Introduced atomic special tokens
  - [CLS]: First token used for sent pair classification
  - [SEP]: Separator between sentences
  - [MASK]: Mask token
- Pretrained on BooksCorpus ($800m$ tokens) + English Wikipedia ($2.5b$ tokens)
  - Batch size 256 seqs of 512 tokens: $128k$ tokens per batch
  - $1m$ updates: 40 epochs over $3.3b$ tokens
  - Adam with weight decay, linear LR warmup step $10k$, dropout 0.1, gelu activation
- Other tricks: E.g., train on length-128 for 90% steps first
Illustration of BERT

IsNext
barked
ran

Transformer
(Vaswani et al., 2017)
Using Pretrained BERT

- Add a light-weight classification layer for each task
- **Finetune.** Instead of holding BERT parameters frozen, jointly optimize them all along with added layer

- Not very sensitive to input representation, sensible choices
  - Sentence pair: “[CLS] s1 [SEP] s2”
  - Single sentence: “[CLS] s”

- Importantly, often just works with one of a small number of hyperparameter configurations!
  - Batch size: 16, 32, dropout: 0.1, learning rate (Adam): $5 \times 10^{-5}$, $3 \times 10^{-5}$, $2 \times 10^{-5}$, 3-10 epochs
BERT-Based Architectures for Downstream Tasks

(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(c) Question Answering Tasks: SQuAD v1.1

(b) Single Sentence Classification Tasks: SST-2, CoLA

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER
The Era of Pretrained Language Models

GPT: Transformer LM (use last hidden state)

The success of BERT started an era of large-scale pretrained language models, in particular trained by MLM

- RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2019), T5 (Raffel et al., 2019), ...
The Unreasonable Effectiveness of Pretrained Transformers

- Game of scale/engineering: Marginal changes in approach/architecture/loss
  - T5 has 5 billion parameters, trained on 1 trillion tokens
  - GLUE score: human 87.1, transformer 90.3 (T5)
    - Recall: WNLI seems to require common sense. Human accuracy 95.9. Transformer accuracy 95.9 (ERNIE).

- Weird situation
  - Before: How can we make it work?
  - Now: *How can it work so well??*

- Explosion of research around pretrained transformer LMs/MLMs
  - What information does a pretrained MLM contain? How can it even seem to solve the Winograd challenge?
  - How can we make training more data-efficient (e.g., ELECTRA (Clark et al., 2020))?
  - How can we train multi-lingual transformers effectively?
  - And many more