Neural Sequence Models

He He

New York University

November 10, 2021

Logistics

- Tutorial on HW4 (constituent parsing) by Udit Arora (TBA)
- Next three weeks: deep learning methods and applications
- Guest lecture on 12/1:
  How far have we come in giving our NLU systems common sense?

Verneek

I'm co-founder of Verneek, a new deep-tech AI startup based in NYC with the mission of enabling anyone to make data-informed decisions. We are now hiring for various roles. It's the best time to join us as we are putting together our core technical team. **Apply here**.

I work on building AI systems that can **comprehend human language**, in a deep manner, where they can show basic **commonsense reasoning** capabilities and "**explain**" themselves! I mainly model language in terms of '**events**' and their temporal and **causal** relations, through which we can build causal networks that predict **what happens next**! The applications of my work range from **storytelling** to **vision & language**.

- Project presentation on 12/8: 3 minutes + 1 minute Q&A (10%)

Modular approaches to NLP

Example: **phrase-based machine translation**

When I look at an article in Russian, I say: This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode. —Warren Weaver

Noisy-channel model:

Fr → En

$x \rightarrow y$

tgt En

LM

translation model

$$p(y \mid x) = \frac{p(y)p(x \mid y)}{\sum_y p(y)p(x \mid y)}$$

src Fr (noised En)

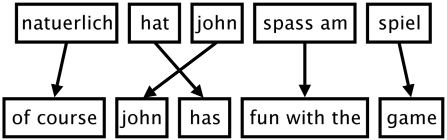
P(le | the)

Word alignment: $p(x \mid y) = \sum_a p(x, a \mid y)$ → P(Fr word | aligned En words)

P(1, 1, 7)

Le	Programme	a	ate	mis	en	application
The	Program	has	been	implemented		

Example: phrase-based MT pipeline



1. Preprocessing: tokenization, truecasing, cleaning
2. Train a (n-gram) language model on the target data
3. Train the translation model
   3.1 Estimate word alignment using EM
   3.2 Extract and score phrase pairs from aligned examples
   3.3 Learn the reordering model
4. Learn a linear model to score hypothesis: features include translation score, LM score, reordering score etc.

Where do we use domain-specific knowledge?

End-to-end approaches to NLP

Sequence-to-sequence models (aka encoder-decoder models):

- Directly model $p(y \mid x)$ with minimal assumption on the sequence structure
- Encoder: $\phi_{\text{enc}} \colon \mathcal{X} \to \mathbb{R}^d$
- Decoder: $\phi_{\text{dec}} \colon \mathbb{R}^d \to \mathcal{Y}$

Extremely flexible framework:

- Summarization: document to summary
- Open-domain dialogue: context to response
- Parsing: sentence to linearized trees
- In general: text to text

A simple implementation of seq2seq

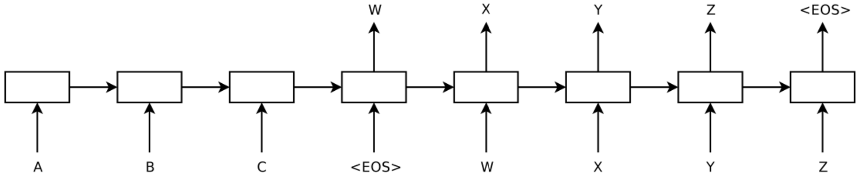


Figure: Sequence to Sequence Learning with Neural Networks [Sutskever+ 2014]

- Encoder/decoder: uni-directional multi-layer LSTM
- Large improvement when the input sequence is reversed
- Outperforms phrase-based MT systems: 34.8 vs 33.3 (on WMT'14 En-Fr)

Seq2seq for constituent parsing

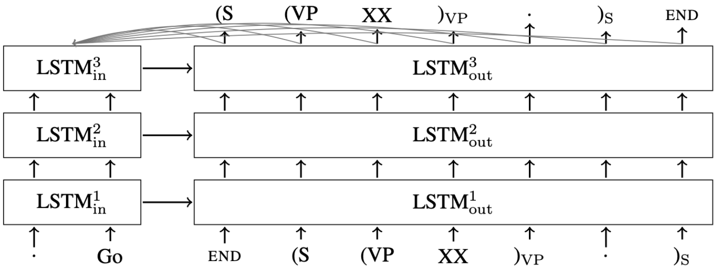
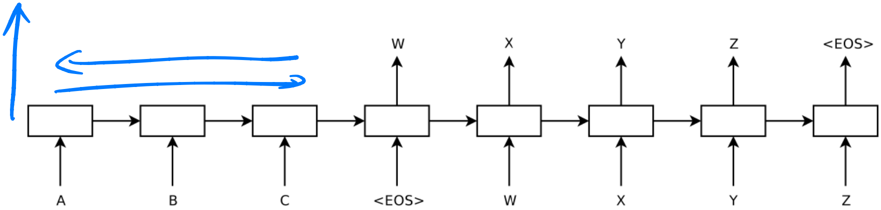


Figure: Grammar as a Foreign Language [Vinyals+ 2015]

- Text to linearized parse trees (no binarization)
- Seq2seq enhanced with attention mechanism (later)
- Matches result from BerkelyParser

Table of Contents

Variants of RNN-based seq2seq architectures

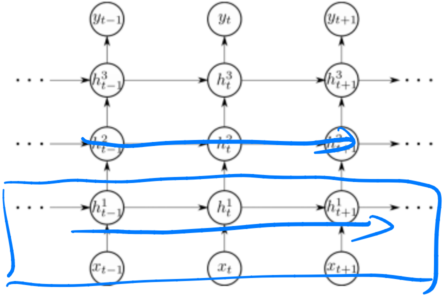


- Basic recurrent unit: vanilla RNN, LSTM, GRU
- Number of layers
- Uni-directional / bi-directional
- Decoder input/output embedding sharing
- Attention mechanism

Multiple layers

Multi-layer RNN (aka stacked RNN):

- Previous layer's outputs are inputs to the next layer
- Use the last layer's output as the input embedding



Pros: "deep" models work better in practice

Cons: longer runtime

Decoder embedding sharing

Input layer: embed previous word y_{i-1}

$$y_{i-1} \mapsto \underbrace{W_{\text{in}}}_{|V|\times d}\phi_{\text{one-hot}}(y_{i-1})$$

Output layer: distribution over the next word y_i

$$h_i \mapsto \text{softmax}(\underbrace{W_{\text{out}}}_{|V|\times d} h_i + b)$$

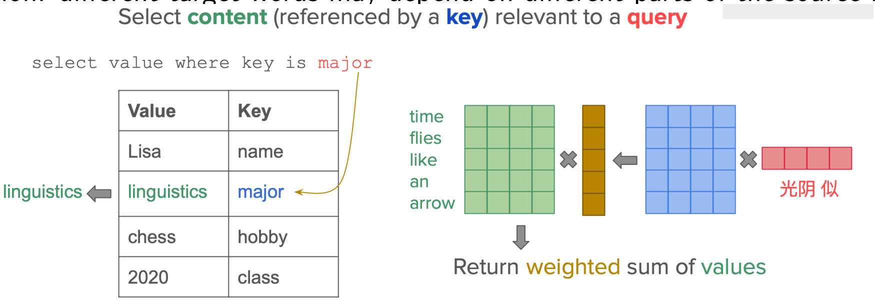
$$\underbrace{W_{out}[y_i, :]}_{\text{emb}(y_i)} \cdot h_i + b$$

Decoder input/output embedding sharing (aka weight tying)

- $W_{\text{in}} = W_{\text{out}}$ (what is the implicit constraint?)
- Intuition: the inner product of h_i and the word embedding of y_i indicates how likely y_i is.
- Worth considering if you don't have lots of data or want to reduce model size

Attention mechanism

Motivation: different target words may depend on different parts of the source input



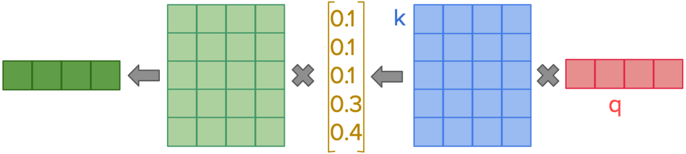
Value	Key
Lisa	name
linguistics	major
chess	hobby
2020	class

Attention is a pooling/aggregation mechanism:

- Encoder states: a memory of *key-value* pairs $(k_1, v_1), \ldots, (k_n, v_n)$.
- Decoder states: a *query* to retrieve from the memory by matching the *keys*.
- Output from the memory: a weighted combination of the *values*.

Attention mechanism

- Query / Values: sentence or word embeddings
- Keys: projections of values



- How likely is q matched to k_i: score $a_i = \alpha(q, k_i)$
- Normalize scores to get attention weights: $b_i = \mathrm{softmax}(a)[i]$
- Output weight combination of values in the memory: $o_i = \sum_{i=1}^{n} b_i v_i$
- In matrix form: attention(Q, K, V) (rows are corresponding vectors)

Common attentions

Design the similarity function between queries and keys: $a_i = \alpha(q, k_i)$

Dot-product attention

$$\alpha(q, k) = q \cdot k$$

MLP attention

$$\alpha(q, k) = u^T \tanh(W[q; k])$$

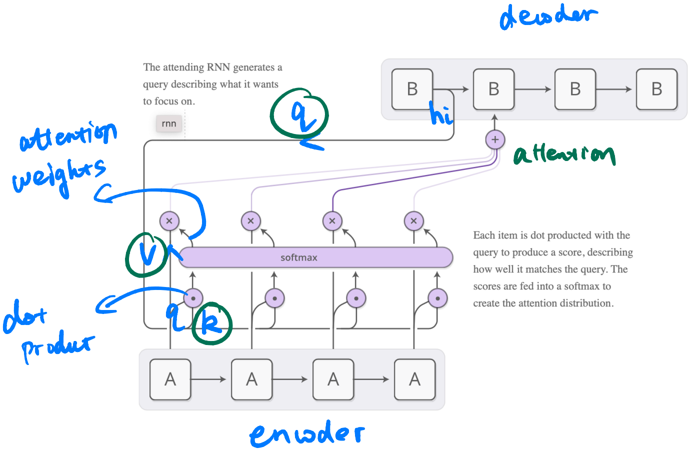
Multi-head attention

h sets of Q, K, V matrices

$$\text{head}_i = \text{attention}(QW_i^Q, KW_i^K, VW_i^V)$$
$$\text{output} = [\text{head}_1; \ldots; \underline{\text{head}_h}]W^O$$

Compute attention with h linear projections of (Q, K, V).

Attention in encoder-decoder models



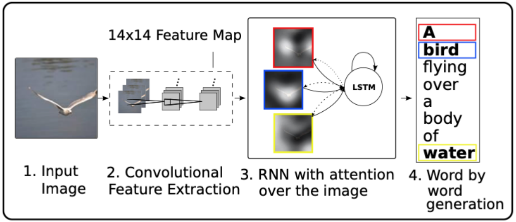
Without attention: $p(y_i \mid y_{<i}, x) \propto f(y_{i-1}, h_{i-1})$

With attention: $p(y_i \mid y_{<i}, x) \propto f(y_{i-1}, h_{i-1}, c_{i-1})$

Applications of attention

In general, adding attention often improves results in encoder-decoder models.

Visual attention:



Use caution with interpretation

Attention is not Explanation [Jain+ 2019]

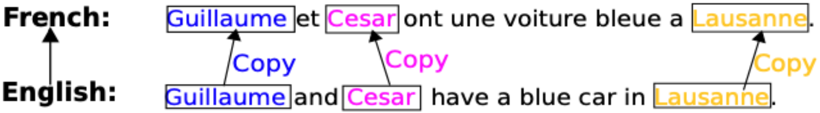
Attention is not not Explanation [Wiegreffe+ 2019]

Learning to Deceive with Attention-Based Explanations [Pruthi+ 2020]

Copy mechanism

Motivation: reuse words in the source

Unknown words in MT:



Dialogue, summarization:

I: `Hello Jack, my name is Chandralekha.`
R: `Nice to meet you, Chandralekha.`

I: `This new guy doesn't perform exactly as we expected.`
R: `What do you mean by "doesn't perform exactly as we expected"?`

Copy mechanism

Interpolate two distributions:

over |V| over words in input

$$p(y_i \mid x, y_{<i}) = \lambda_{\text{gen}} p_{\text{gen}}(y_i \mid x, y_{<i}) + (1 - \lambda_{\text{gen}}) p_{\text{copy}}(y_i \mid x, y_{<i})$$

- p_{gen}: distribution over words in the vocabulary
- p_{copy}: distribution over words in the source

Design decisions:

- Learned (function of the input) vs fixed λ_{gen} $= f(h_i)$
- p_{copy}: use attention weights or compute from a separate model

Application of the copy mechanism

Most successful in abstractive summarization

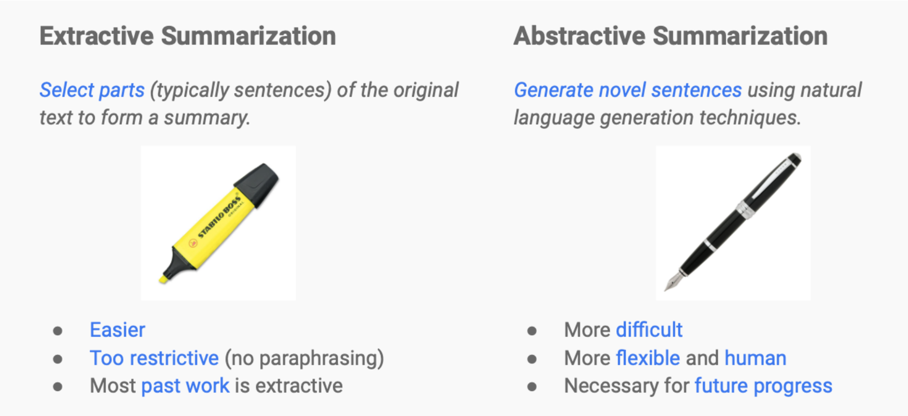


Figure: Slides from Abigail See

Application of the copy mechanism

Most successful in abstractive summarization

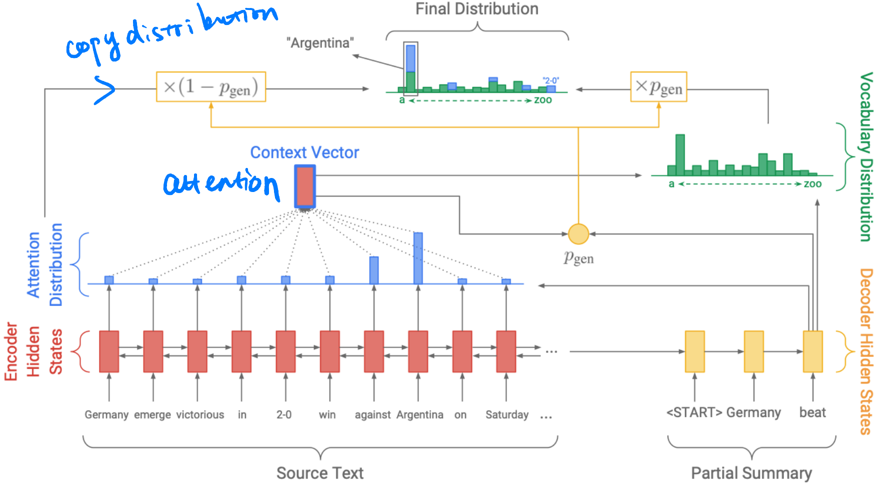
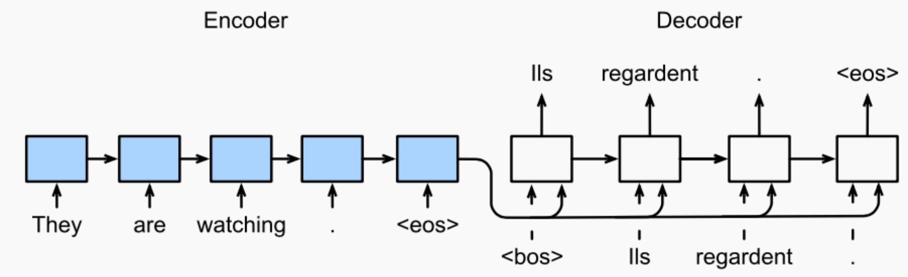


Figure: Pointer-Generator network [See+ 2016]

Table of Contents

Training



MLE:

$$\max_{\theta} \sum_{i=1}^{N} \log p(y^{(i)} \mid x^{(i)}; \theta)$$

softmax(LSTM(ht, ct))

$$= \max_{\theta} \sum_{i=1}^{N} \sum_{t=1}^{T} \underbrace{\log p(y_t^{(i)} \mid x^{(i)}, y_{1:t-1}^{(i)}; \theta)}_{\text{decoder output}}$$

auto-regressive model

Argmax decoding

Argmax decoding (aka MAP decoding):

$$\hat{y} = \arg\max_{y \in \mathcal{Y}^n} p(y \mid x; \theta)$$

- Return the most likely sequence
- $\mathcal{Y}$ is the vocabulary size for text generation
- Exact search is intractable when scores aren't locally decomposable

Approximate search:

- **Greedy decoding**: return the most likely symbol at each step

$$y_t = \arg\max_{y \in \mathcal{Y}} p(y \mid x, y_{1:t-1}; \theta)$$

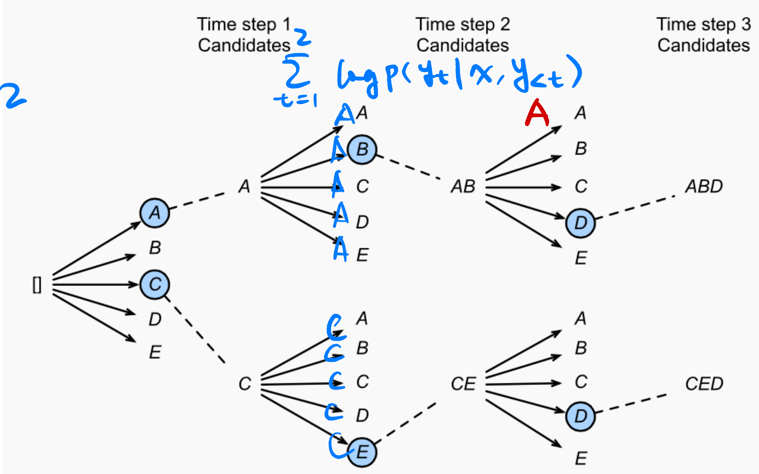
Approximate MAP decoding: beam search

Beam search: maintain k highest-scored *partial* solutions at any time

$|V| = 5$

beam size $= 2$

$\sum_{t=1}^{2} \log p(y_t | x, y_{<t})$



Is MAP the right decoding objective?

High likelihood can be correlated with low quality outputs.

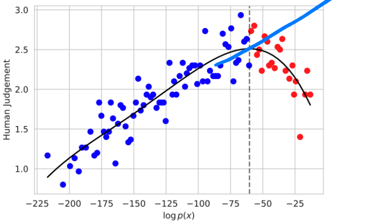


Figure: Samples from an LM [Zhang+ 2020]

In practice, argmax decoding has been observed to lead to

- Repetitive generations, e.g.

  "..., was conducted by researchers from the Universidad Nacional Autonoma de Mexico (UNAM) and the Universidad Nacional Autonoma de Mexico (UNAM/Universidad Nacional Autonoma de Mexico/Universidad Nacional Autonoma de Mexico/Universidad Nacional Autonoma..."

- Degraded generations with large beam size in MT

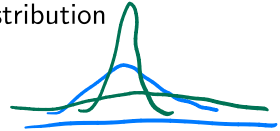
  3-5

Sampling-based decoding

Directly sampling from $p(y \mid x; \theta)$ often produces non-sensical sentences:

> They were cattle called Bolivian Cavalleros; they live in a remote desert uninterrupted by town, and they speak huge, beautiful, paradisiacal Bolivian linguistic thing.

Tempered sampling: change the concentration of the distribution



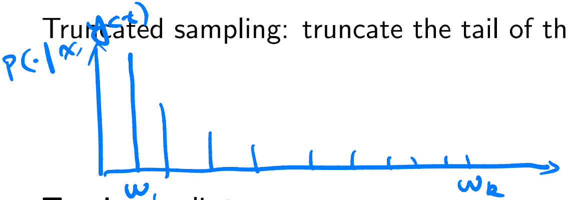
$$p(y_t \mid x, y_{1:t-1}; \theta) \propto \exp \underbrace{\left(s_\theta(y_t, x, y_{1:t-1})\right)}_{\text{score of } y_t}$$

$$q(y_t \mid x, y_{1:t-1}) \propto \exp \left(s_\theta(y_t, x, y_{1:t-1})/T\right) \quad \text{where } T \in (0, +\infty)$$

- What happends when $T \to 0$ and $T \to +\infty$?
- Does it change the rank of y according to likelihood?
- Typically we chooose $T \in (0, 1)$.

Sampling-based decoding

Truncated sampling: truncate the tail of the distribution



Top-k sampling:

80%

$$q(y_t \mid x, y_{1:t-1}) \propto p(y_t \mid x, y_{1:t-1}; \theta)\mathbb{I}(r(y_t) \leq k)$$

where $r(y)$ for $y \in \mathcal{Y}$ returns the rank of y by $p(y_t \mid x, y_{1:t-1}; \theta)$ in descending order.

Top-p sampling (aka nucleus sampling):

$$q(y_t \mid x, y_{1:t-1}) \propto p(y_t \mid x, y_{1:t-1}; \theta)\mathbb{I}(\sum_{i=1}^{r(y_t)} f(i) \leq p)$$

where $f(i)$ for $i \in |\mathcal{Y}|$ returns i-th highest $p(y_t \mid x, y_{1:t-1}; \theta)$.

Decoding in practice

Rule of thumb:

- Use beam search with small beam size for tasks where there exists a correct answer, e.g. machine translation, summarization
- Use top-k or top-p for open-ended generation, e.g. story generation, chit-chat dialogue, continuation from a prompt

Table of Contents

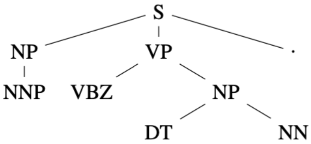
Applications

Text generation: MT, summarization, chit-chat dialogue, image caption, story generation etc.

Structured prediction:

- Parsing

John has a dog . $\rightarrow$



John has a dog . $\rightarrow$ (S (NP NNP $)_{\mathrm{NP}}$ (VP VBZ (NP DT NN $)_{\mathrm{NP}}$ $)_{\mathrm{VP}}$. $)_{\mathrm{S}}$

- Text-to-SQL

Evaluation

Evaluate translations:

- **Reference 1** It is a guide to action that ensures that the military will forever heed Party commands.
- **Reference 2** It is the guiding principle which guarantees the military forces always being under the command of the Party.
- **Candidate 1** It is a guide to action which ensures that the military always obeys the commands of the party.
- **Candidate 2** It is to insure the troops forever hearing the activity guidebook that party direct.

Task: given the reference(s) of each source sentence, evaluate the quality of the generated sequences.

Main idea: good generations should have high overlap with the reference.

BLEU: n-gram precision

First try: n-gram precision (x: input, c: candidate, r: reference)

$$p_n = \frac{\sum_{(x,c,r)} \sum_{s \in \text{n-gram}(c)} \mathbb{I}\left[s \text{ in } r\right]}{\sum_{(x,c,r)} \sum_{s \in \text{n-gram}(c)} \mathbb{I}\left[s \text{ in } c\right]}$$

BLEU: n-gram precision

First try: n-gram precision (x: input, c: candidate, r: reference)

$$p_n = \frac{\sum_{(x,c,r)} \sum_{s \in \text{n-gram}(c)} \mathbb{I}\left[s \text{ in } r\right]}{\sum_{(x,c,r)} \sum_{s \in \text{n-gram}(c)} \mathbb{I}\left[s \text{ in } c\right]}$$

Problem: matching only a few words in the reference(s)

Candidate	the the the the the the the
Reference 1	The cat is on the mat
Reference 2	There is a cat on the mat

unigram precision = ?

Solution: clip counts to maximum count in the reference(s)

BLEU: n-gram precision

Given p_n's, we need to combine n-gram precisions.
Weighted average? Problem: precision decreases roughly exponentially with n.

Solution: geometric mean (when $w_n = 1/n$)

$$\exp\left(\sum_{i=1}^{n} w_n \log p_n\right)$$

Problem with precision:

Candidate	of the
Reference 1	It is the guiding principle which guarantees the military forces always being under the command of the Party.
Reference 2	It is the practical guide for the army always to heed the directions of the party.

What are problems with recall with *multiple* references?

BLEU: brevity penalty

A good translation must match the reference in:

- word choice: captured by precision
- word order: capture by n-gram
- length: ?

candidate length $C = \sum_{(x,c,r)} \text{len}(c)$

reference length $R = \sum_{(x,c,r)} \arg\min_{a \in \{\text{len}(r_1),\ldots,\text{len}(r_k)\}} |a - \text{len}(c)|$

- Use the reference whose length is closest to the candidate

Brevity penalty $BP = \begin{cases} 1 & \text{if } c > r \\ e^{1-R/C} & \text{if } c \leq r \end{cases}$

- No penalty if $r \leq c$

BLEU

Putting everything together:

$$\text{BLEU} = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

$$\log \text{BLEU} = \min(1 - \frac{R}{C}, 0) + \sum_{n=1}^{N} w_n \log p_n$$

- Both precision and the brevity penalty are computed at the *corpus level*.
- Need smoothing for sentence-level BLEU.
- Good correlation with human evaluation for MT (typically $n = 4$).

ROUGE

Task: given a candidate summary and a set of reference summaries, evaluate the quality of the candidate.

ROUGE-n: n-gram recall

- Encourage content coverage

ROUGE-L: measures longest common subsequence between a candidate and a reference

- Precision $= LCS(c, r)/\text{len}(c)$
- Recall $= LCS(c, r)/\text{len}(r)$
- F-measure $= \frac{(1+\beta^2)RR}{R+\beta^2 P}$
- Doesn't require consecutive match.

Often used for summarization, but human evaluation is still needed.

Automatic evaluation metrics for sequence generation

n-gram matching metrics (e.g. BLEU, ROUGE)

- Measures exact match with reference; interpretable.
- Do not consider semantics.

Embedding-based metrics (e.g. BERTScore)

- Measures similarity to the reference in an embedding space.
- Captures synonyms and simple paraphrases.

However, we also want to measure

- Is the generation correct? e.g. faithfulness (summarization), adequacy (MT).
- Open-ended generation: is the story/dialogue interesting, informative, engaging?

Automatic evaluation metrics for sequence generation

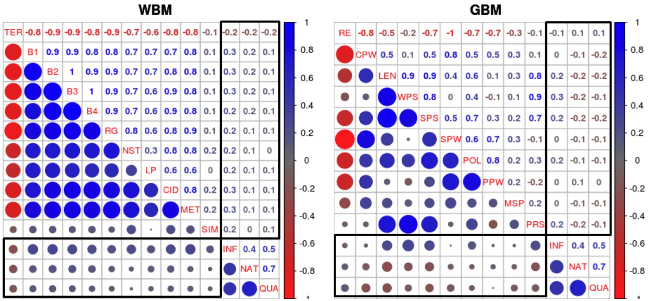


Figure: [Novikova+ 2017]

- Correlation between automatic metrics and human ratings on generation quality
- Left: word-overlap metrics; right: grammar-based metrics
- Overall, low correlation with human ratings

Human Evaluation

- Human or machine generated?

Once upon a time, there lived a pirate. He was the sort of pirate who would rather spend his time chasing away the sharks swimming around his ship than sail to foreign ports in search of booty. He was a good pirate, a noble pirate, an honest pirate. He was a pirate who would rather be at home with his wife and son than out on a ship in the middle of the ocean.

Human Evaluation

- Human or machine generated?

> Once upon a time, there lived a pirate. He was the sort of pirate who would rather spend his time chasing away the sharks swimming around his ship than sail to foreign ports in search of booty. He was a good pirate, a noble pirate, an honest pirate. He was a pirate who would rather be at home with his wife and son than out on a ship in the middle of the ocean.

- Human evaluation can be tricky as the models gets better!
- Pros: more reliable, multifaceted evaluation
- Cons: high variance, misalignment

Evaluation in practice

Evaluation is a key blocker to progress in text generation.

In practice, multiple evaluation methods are needed for reliable results:

- Held-out NLL/perplexity: how close are $p_\theta(y \mid x)$ and $p(y \mid x)$?
- Automatic evaluation: how close are the candidate generation and the reference(s)?
- Human evaluation: task-specific criteria, e.g. grammaticality, coherence, correctness etc.
  - Annotator may need to be trained
  - Need to report annotator agreement
- *Show the outputs!*