Sequence Models: seq2seq, LSTMs

CPSC 440/550: Advanced Machine Learning

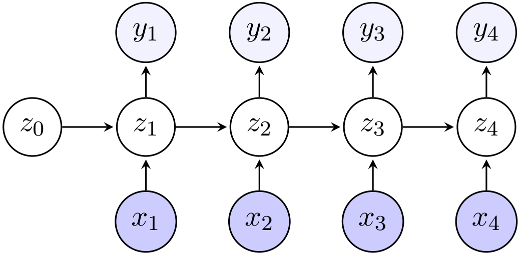
`cs.ubc.ca/~dsuth/440/24w2`

University of British Columbia, on unceded Musqueam land

2024-25 Winter Term 2 (Jan–Apr 2025)

Last time

- Recurrent neural networks (RNNs): a discriminative model for sequences



  - z_0 is a fixed vector
  - $z_t = f(x_t, z_{t-1})$ computed deterministically with some neural net
  - For discrete y, $y_t \sim \mathrm{Cat}(g(z_t))$ is a multiclass classification problem
  - Could also use another y distribution, e.g. $\mathcal{N}(g(z_t), \sigma^2)$ for continuous scalar labels
- Maximum likelihood: $\arg\min_{f,g} -\sum_{i=1}^{n} \sum_{t=1}^{T^{(i)}} \log p(y_t^{(i)} \mid x_{1:t}^{(i)}, f, g)$
  - Optimization challenges, especially for long sequences
- Variants: stack multiple hidden layers (deep RNN), bi-directional, ...

Outline

Motivating problem: Machine translation

- In machine translation:
  - Input is text in language A
  - Output is text in language B with same meaning

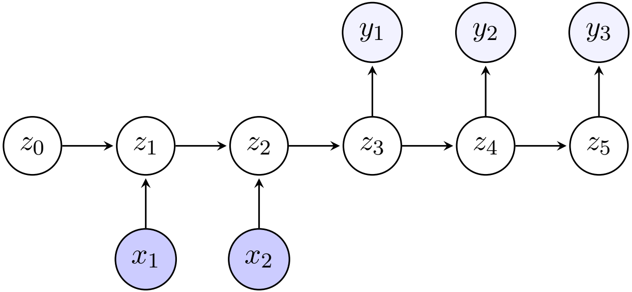
This course is intended as a second or third university-level course on machine learning, a field that focuses on using automated data analysis for tasks like pattern recognition and prediction.

Ce cours est conçu comme un cours de deuxième ou troisième niveau universitaire sur l'apprentissage automatique, un domaine qui se concentre sur l'utilisation de l'analyse de données automatisée pour des tâches telles que la reconnaissance de formes et la prédiction.

- Compared to per-pixel labeling:
  - Input, output sequences probably have different lengths and different “order”
  - It’s not just “which French word corresponds to this English word”
  - We probably don’t know the output length

Sequence-to-sequence RNNs

- Sequence-to-sequence (seq2seq) models encode a sequence, then decode:
- Each encoding sstep takes a word as input, and outputs nothing
- Each decoding step takes no input and outputs a word
  - Different (tied) parameters inside the encoder and the decoder



- Switch from encoder to decoder when the input sequence ends
- When to stop the decoder? When it generates a special `<EOS>` word

Sequence-to-sequence loss function

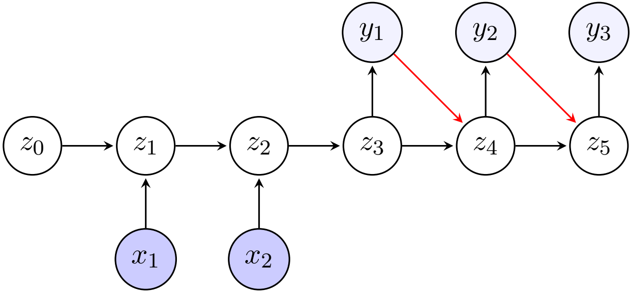
- The sequence-to-sequence loss looks like

$$\arg\min_{f,g} - \sum_{i=1}^{n} \sum_{t=1}^{\left|y^{(i)}\right|} \log p(y_t^{(i)} \mid x^{(i)}, f, g)$$

- This is trying to get each label right, not the "whole sequence"
- For example, translating in a slightly different way might "break everything"

Variant: dependent predictions

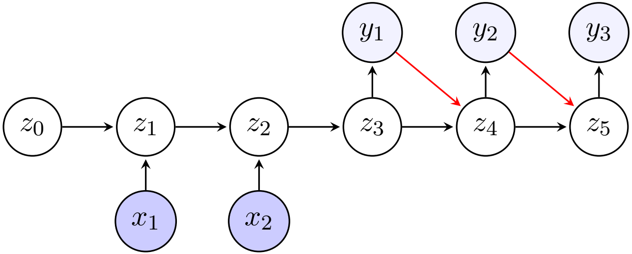
- Standard RNN model assumes $y_t \perp\!\!\!\perp y_{t+1} \mid z_t$
- This makes inference easy
- But our prediction for y_t "forgets" what we picked for y_{t-1}
- Variant: add edges like this



- Fine at training time, since we know the ys
- Sampling is also still fine
- Decoding the mode is much harder

Beam search

- In a model with dependent predictions:



- Could try to use complicated dynamic programming from UGMs for decoding
- Usual alternative: beam search, a heuristic that usually works okay
  - Run encoding as normal (it's deterministic in this model)
  - Take the best B (beam size) out of V (vocab size) choices for y_1
    - Best: ones with highest probability under this model
  - Consider all of the BV choices for (y_1, y_2) that start from the beam, keep the best B
  - Stop when you hit `<EOS>` for everything in the beam or get bored

Outline

What level to model at?

- We've been talking about x_t as a word
  - Is "British Columbia" one word or two?
  - How many words is "New York–New Jersey border"?
  - What about 地下铁路 (dìxià tiělù) = "subway"?

    地(ground) + 下(under) = 地下(underground)

    铁(iron) + 路(road) = 铁路(railroad)
- Should we really have every word from every language separate in the vocabulary?
- What about typos? Uncommon names? Slang we haven't seen before?

- Character-level modeling is way more flexible, but makes sequences really long
  - Also there are 74,000+ Chinese characters and 3,000+ emoji...

- Byte-level modeling: put everything in UTF-8, then we only need 256 characters!
  - Sequences are even longer...

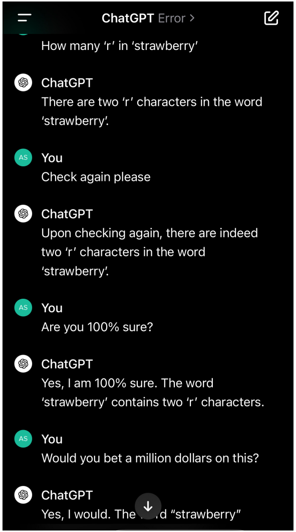
- Whichever we pick, we usually call the modeling unit a token

Byte-Pair Encoding

- Word-level: huge vocabulary, shorter sequences, big out-of-vocab problem
- Character-level: smaller vocabulary, longer sequences, still out-of-vocab problem
- Byte-level: small vocabulary, even longer sequences, works for any UTF-8 string

- Usual in-between: variant of Byte-Pair Encoding
  - Start out with 256 single bytes
  - Repeat: for the most commonly co-occurring pair `A B`, make a new token `AB`
  - Stop at a desired vocab size (a few tens of thousands)
- Often special treatment for punctuation, spaces, etc
  - Do we really want `dog.` instead of `dog, .`?
- Often special treatment for numbers, . . .

Tokenization can be really important!



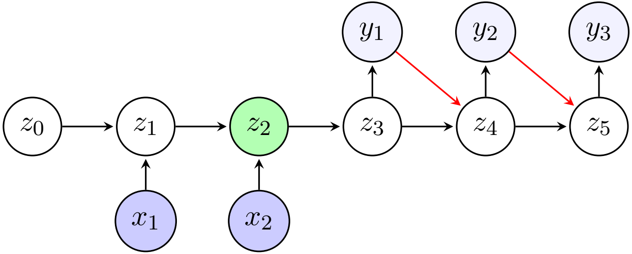
https://community.openai.com/t/incorrect-count-of-r-characters-in-the-word-strawberry/829618

https://www.reddit.com/r/softwaregore/comments/187tvwr/cenya_moment/

Outline

Exponential “forgetting” in RNNs

- In a seq2seq RNN, whole input sequence gets encoded into a single hidden state



- Everything has to get squeezed into this one fixed-dim vector
- Info from a long time ago went through $f(x_{t-1}, f(x_{t-2}, f(x_{t-3}, \ldots)))$
  - Like gradient explosion/vanishing, “information” can easily vanish too
- Three major approaches to avoid this:
  - Special structures on z with state-space models (S4, H3, Mamba, . . .)
  - Adding skip connections (leads to attention, . . . , soon)
  - Refining the update mechanism (LSTMs, GRUs, . . . , next)

State Space Models

bonus!

- A class of models using fancy math to make sure information "sticks around" in z
  - Often continuous-state and even continuous-time
  - Requires restricting the evolution of z a lot (usually linear)
  - Complex (deep) mapping between latent z and observations/labels
- Lots of effort over the past 4-ish years to make these work for text

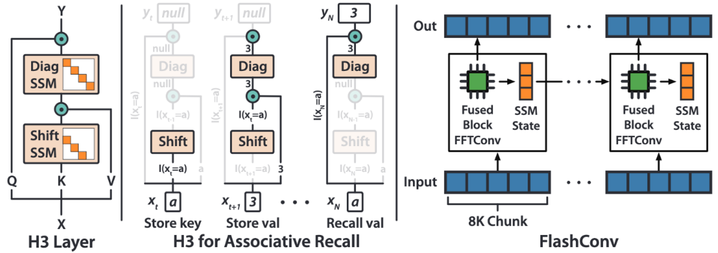
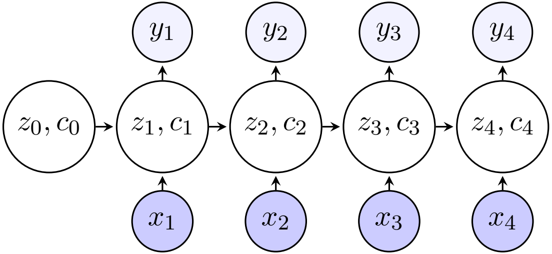


Figure 1: Left: H3 stacks two discrete SSMs with shift and diagonal matrices and uses multiplicative interactions between input projections and their outputs to model comparisons between points in a sequence. Middle: H3 can perform associative recall—which is easy for attention, but not existing SSMs. Right: FLASHCONV uses a new state-passing algorithm over fused block FFTConv to increase hardware efficiency of SSMs, allowing H3 to scale to billion-parameter models.

Long short-term memory (LSTM)

- Nets' "long-term memory" is in weights, "short-term memory" in activations
- The problem of "forgetting" is that it's hard to keep things in short-term memory
- LSTMs add an explicit mechanism to "write stuff down for later":

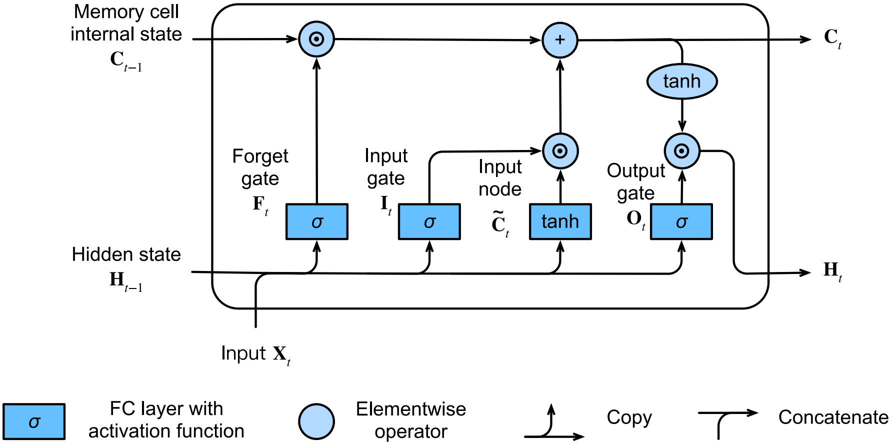


- Need a way to decide when to save to, read from, or clear memory cells
- In a regular program, this would be some kind of `if` statement based on inputs
  - But how can we learn that with gradient methods?

LSTM unit

- Idea: represent "hard decisions" as binary values in $\{0, 1\}$
  - To learn them, we'll turn them into "soft decisions" in $[0, 1]$
- Forget gate asks "should I reset the old memory?"
  - If $F_t = 0$, we forget the old value; if $F_t = 1$, we remember it
  - We'll access memory as $c_{t-1} \odot F(x_t, z_{t-1})$; $\odot$ is elementwise product
  - F is a simple net: $\sigma(W_{f,x} x_t + W_{f,z} z_{t-1} + b_f)$ where σ is elementwise logistic sigmoid
  - We're handling multiple cells at once, with separate decisions for each
- Input gate asks "should I add something to the memory?"
  - If $I_t = 0$, we don't add anything new; if $I_t = 1$, we do
  - Set $c_t = F(x_t, z_{t-1}) \odot c_t + I(x_t, z_{t-1}) \odot V(x_t, z_{t-1})$
  - I is a simple net: $\sigma(W_{i,x} x_t + W_{i,z} z_{t-1} + b_i)$ where σ is logistic sigmoid
  - V is a simple net: $\tanh(W_{v,x} x_t + W_{v,z} z_{t-1} + b_v)$
- Output gate asks "should I read from memory?"
  - If $O_t = 0$ we don't read out from memory; if $O_t = 1$, we do
  - Output value of the cell is $z_t = O(x_t, z_{t-1}) \odot \tanh(c_t)$
  - O is a simple net: $\sigma(W_{o,x} x_t + W_{o,z} z_{t-1} + b_o)$ where σ is logistic sigmoid

LSTM unit

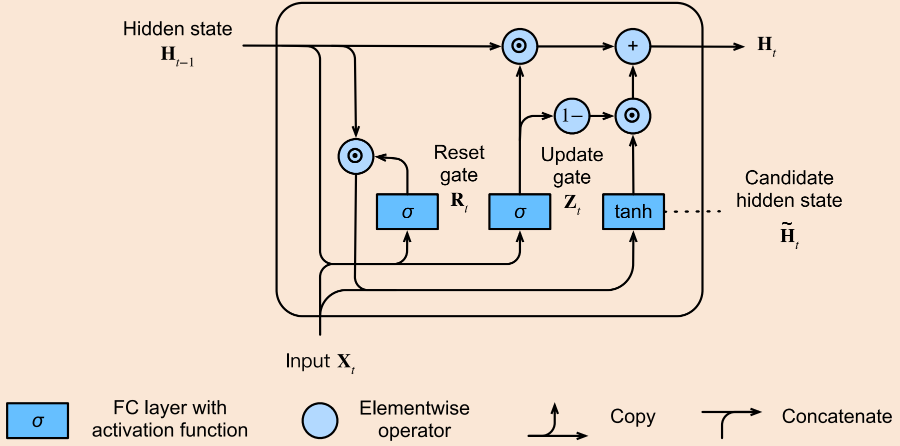


https://d2l.ai/chapter_recurrent-modern/lstm.html

- This whole thing implements $f(x_t, z_{t-1}, c_{t-1})$
- Can still make deep RNNs, bidirectional RNNs, etc etc with these

Gated Recurrent Unit (GRU)

bonus!



https://d2l.ai/chapter_recurrent-modern/gru.html

- Common variant, fewer params with similar performance

Successes of LSTMs

bonus!

- The first model that made RNNs work across many applications:
- Handwriting recognition, especially cursive `https://www.youtube.com/watch?v=mLxsbWAYIpw`
- Speech recognition and text-to-speech (Google, Apple, Amazon c. 2015-17)
- Machine translation (Google, Facebook c. 2016)
- iPhone autocorrect (c. 2016)
- AIs for Dota 2 (OpenAI 2018), Starcraft 2 (DeepMind 2019), . . .

PixelRNN

- PixelRNN model (2016): decompose

$$p(\text{image}) = p(\text{pixel 1})p(\text{pixel 2} \mid \text{pixel 1})p(\text{pixel 3} \mid \text{pixel 1}, \text{pixel 2}) \cdots$$

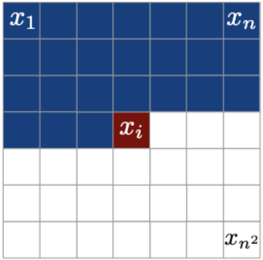


Figure 1. Image completions sampled from a PixelRNN.

https://arxiv.org/abs/1601.06759

  - Implements $p(\text{pixel} \mid \text{context})$ with an LSTM
- Gets exact likelihoods (unlike VAEs), can use discrete pixel values, good model
- Relatively slow (process pixel-by-pixel) and order really matters

Outline

Encoding-decoding for different data types

- seq2seq models use separate “models” for encoding and decoding
- Can think of as mapping sentence to vector, followed by vector to sentence

- VAEs use separate “models” for encoding and decoding
- Can think of as mapping image to vector, followed by vector to image

- . . . we can mix-and-match!

Image captioning model

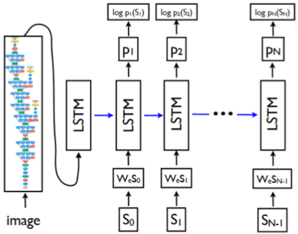


Figure 3. LSTM model combined with a CNN image embedder (as defined in [12]) and word embeddings. The unrolled connections between the LSTM memories are in blue and they correspond to the recurrent connections in Figure 2. All LSTMs share the same parameters.

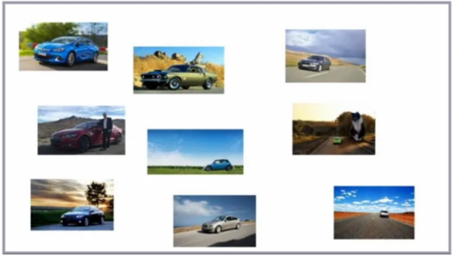
Figure 5. A selection of evaluation results, grouped by human rating.

https://arxiv.org/abs/1411.4555

- Train the models jointly based on dataset of images + captions

Image captioning

Impressive, right? Not so fast, says Efros. "If you go and look for cars on the internet," he points out, "that description applies to pretty much all of those images."



https://mathwithbaddrawings.com/2017/10/18/5-ways-to-troll-your-neural-network/

Image captioning

"a car parked on the side of the road"

"a car parked on the side of the road"

https://mathwithbaddrawings.com/2017/10/18/5-ways-to-troll-your-neural-network/

Image captioning: PDF to $\LaTeX$

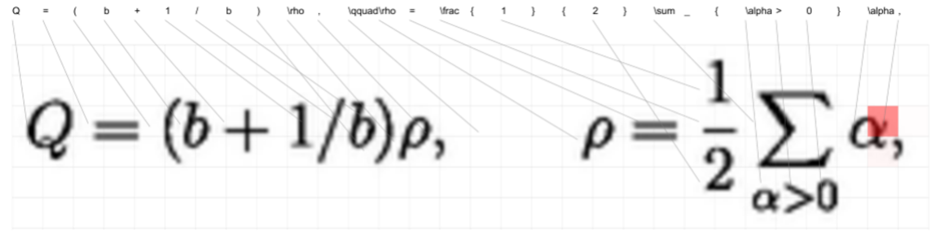
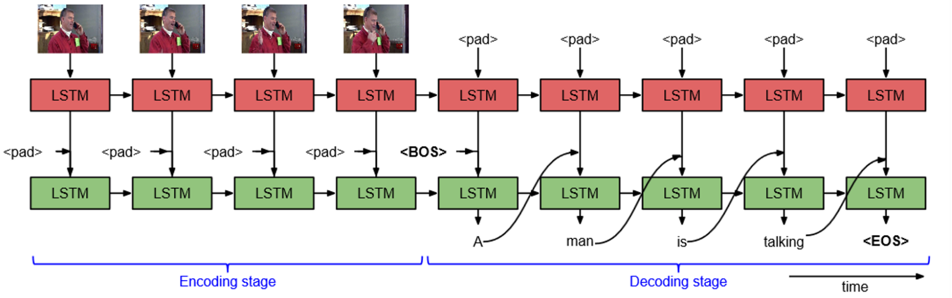


Figure 1: Example of the model generating mathematical markup. The model generates one LaTeX symbol y at a time based on the input image $\mathbf{x}$. The gray lines highlight $H' \times V'$ grid features after the CNN $\mathbf{V}$ and RNN Encoder $\tilde{\mathbf{V}}$. The dotted lines indicate the center of mass of α for each word (only non-structural words are shown). Red cells indicate the relative attention for the last token. See `http://lstm.seas.harvard.edu/latex/` for a complete interactive version of this visualization over the test set.

https://arxiv.org/abs/1609.04938

- Unlike generic captioning, there's a correct answer (though maybe not unique)

Video captioning with LSTMs



http://www.cv-foundation.org/openaccess/content_iccv_2015/papers/Venugopalan_Sequence_to_Sequence_ICCV_2015_paper.pdf

Video captioning with LSTMs

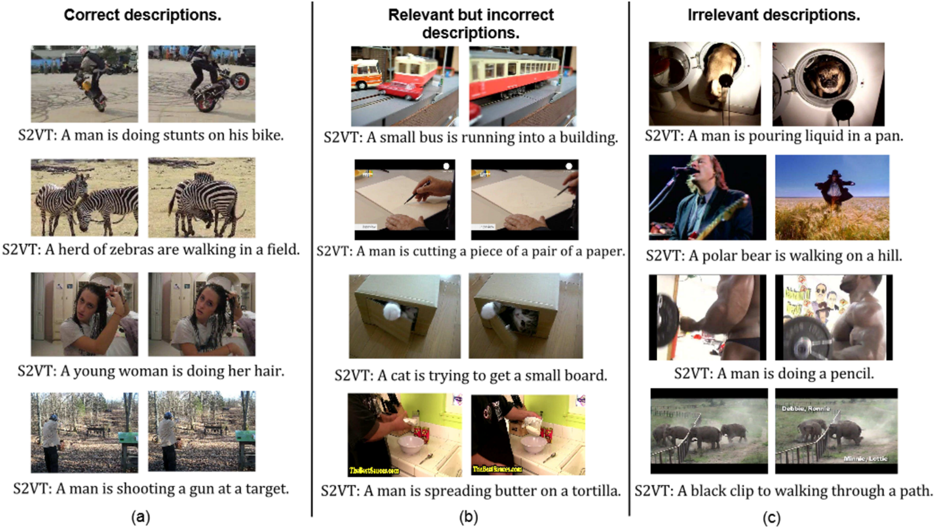


Figure 3. Qualitative results on MSVD YouTube dataset from our S2VT model (RGB on VGG net). (a) Correct descriptions involving different objects and actions for several videos. (b) Relevant but incorrect descriptions. (c) Descriptions that are irrelevant to the event in the video.

Video captioning: Lip reading

https://www.youtube.com/watch?v=5aogzAUPilE

- Unlike generic captioning, there's a correct answer

Summary

- Sequence-to-sequence models: simple reframing, gets outputs of arbitrary length
- Tokenization is important
  - Word-level? Character-level? Byte-level? Usually in between
- LSTMs: modify the state to include "memory cells"
  - Soft "gates" to differentiably read, write, reset
- Multimodal models
  - Encode an image, decode a possible caption

- Next time: Transformers