A TensorFlow Chatbot

CS 20SI:
TensorFlow for Deep Learning Research
Lecture 13
3/1/2017

Announcements

Assignment 3 out tonight, due March 17

No class this Friday: Pete Warden's talk on TensorFlow for mobile

Guest lecture next Friday by Danijar Hafner on Reinforcement Learning

Agenda

Seq2seq

Implementation keys

Chatbot craze

Sequence to Sequence

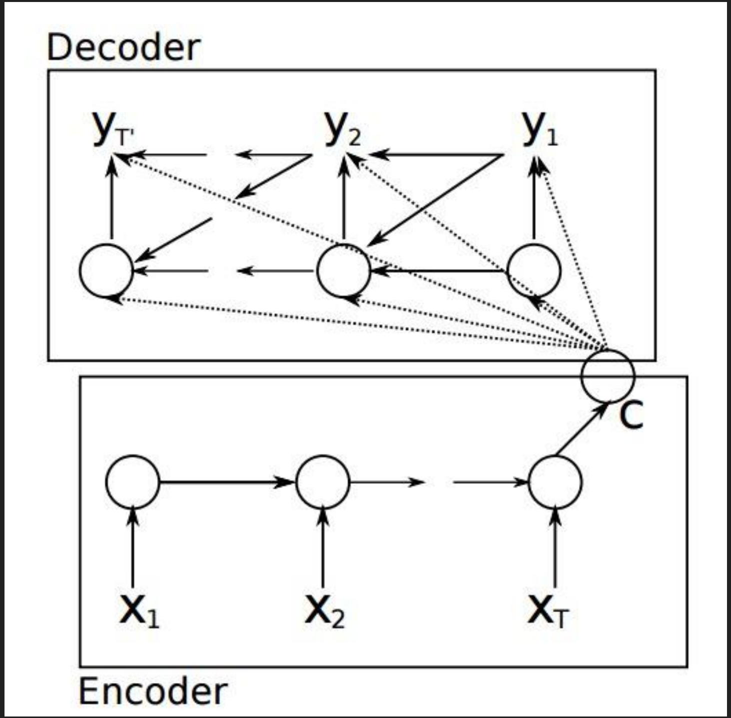
- The current model class of choice for most dialogue and machine translation systems
- Introduced by Cho et al. in 2014 for Statistical Machine Translation (the predecessor of NMT)
- The paper “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation” has been cited 900 times, approx. one paper a day.
- Originally called “RNN Encoder – Decoder”

Sequence to Sequence

Consists of two recurrent neural networks (RNNs):

- Encoder maps a variable-length source sequence (input) to a fixed-length vector
- Decoder maps the vector representation back to a variable-length target sequence (output)
- Two RNNs are trained jointly to maximize the conditional probability of the target sequence given a source sequence

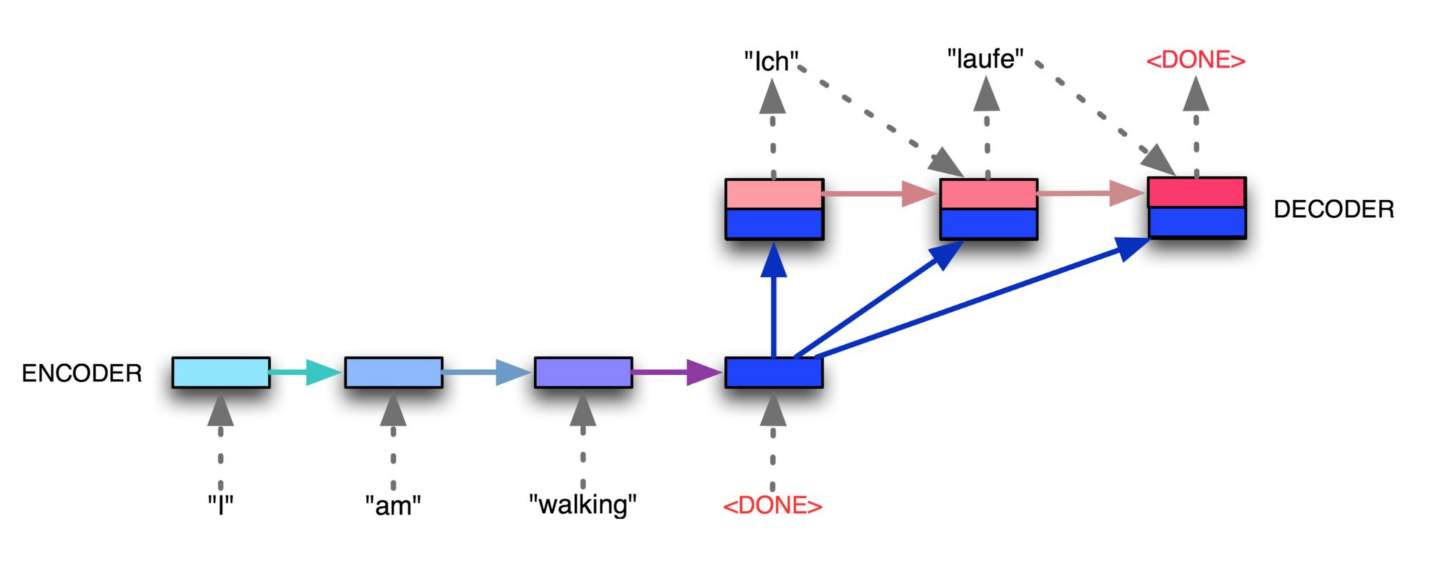
Vanilla Encoder and Decoder



Graph from “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation” (Cho et al.)

Encoder and Decoder in TensorFlow

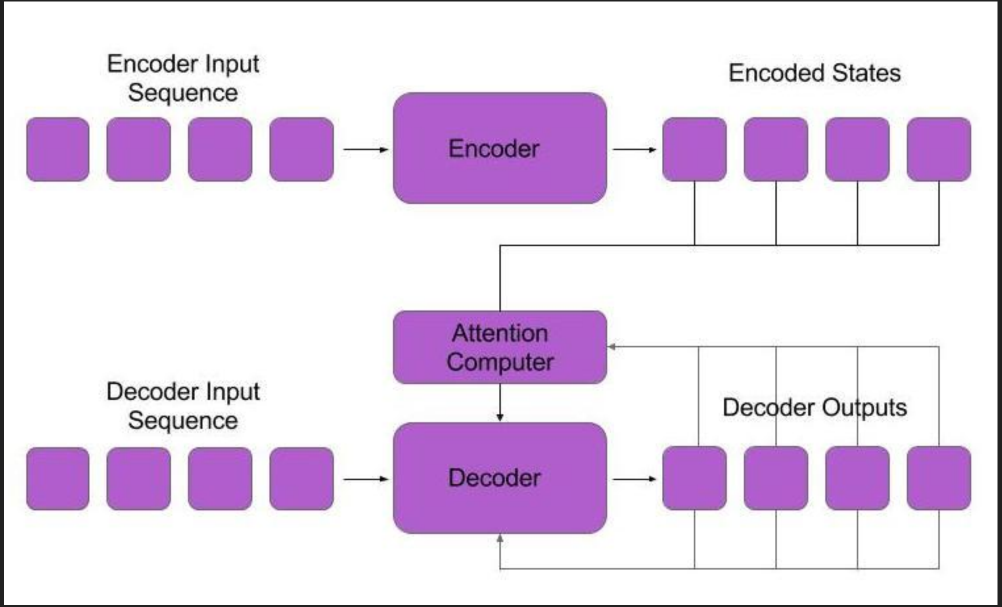
- Each box in the picture represents a cell of the RNN, most commonly a **GRU** cell or an **LSTM** cell.
- Encoder and decoder often have different weights, but sometimes



Graph by Dev Nag

With Attention

- In the vanilla model, each input has to be encoded into a fixed-size state vector, as that is the only thing passed to the decoder.
- Attention mechanism that gives decoder direct access to the input.



Graph by Indico.io blog

Bucketing

- Avoid too much padding that leads to extraneous computation
- Group sequences of similar lengths into the same buckets

Bucketing

- Avoid too much padding that leads to extraneous computation
- Group sequences of similar lengths into the same buckets
- Create a separate subgraph for each bucket

Bucketing

- Avoid too much padding that leads to extraneous computation
- Group sequences of similar lengths into the same buckets
- Create a separate subgraph for each bucket
- In theory, can use for v1.0:

```
tf.contrib.training.bucket_by_sequence_length(max_length,
examples, batch_size, bucket_boundaries, capacity=2 *
batch_size, dynamic_pad=True)
```

- In practice, use the bucketing algorithm used in TensorFlow's translate model (because we're using v0.12)

Sampled Softmax

- Avoid the growing complexity of computing the normalization constant
- Approximate the negative term of the gradient, by importance sampling with a small number of samples.
- At each step, update only the vectors associated with the correct word w and with the sampled words in V'
- Once training is over, use the full target vocabulary to compute the output probability of each target word

On Using Very Large Target Vocabulary for Neural Machine Translation (Jean et al., 2015)

Sampled Softmax

```
if config.NUM_SAMPLES > 0 and config.NUM_SAMPLES < config.DEC_VOCAB:
            weight = tf.get_variable('proj_w', [config.HIDDEN_SIZE, config.DEC_VOCAB])
            bias = tf.get_variable('proj_b', [config.DEC_VOCAB])
            self.output_projection = (w, b)

        def sampled_loss(inputs, labels):
            labels = tf.reshape(labels, [-1, 1])
            return tf.nn.sampled_softmax_loss(tf.transpose(weight), bias, inputs, labels,
                                              config.NUM_SAMPLES, config.DEC_VOCAB)
        self.softmax_loss_function = sampled_loss
```

Sampled Softmax

- Generally an underestimate of the full softmax loss.
- At inference time, compute the full softmax using:

```
tf.nn.softmax(tf.matmul(inputs,  tf.transpose(weight)) + bias)
```

Seq2seq in TensorFlow

```
outputs, states = basic_rnn_seq2seq(encoder_inputs, decoder_inputs, cell)
```

encoder_inputs: a list of tensors representing inputs to the encoder
decoder_inputs: a list of tensors representing inputs to the decoder
cell: single or multiple layer cells

outputs: a list of decoder_size tensors, each of dimension 1 x DECODE_VOCAB corresponding to the probability distribution at each time-step
states: a list of decoder_size tensors, each corresponds to the internal state of the decoder at every time-step

Seq2seq in TensorFlow

```
outputs, states = basic_rnn_seq2seq(encoder_inputs,
                                    decoder_inputs,
                                    cell)
```

encoder_inputs: a list of tensors representing inputs to the encoder
decoder_inputs: a list of tensors representing inputs to the decoder
cell: single or multiple layer cells

outputs: a list of decoder_size tensors, each of dimension 1 x DECODE_VOCAB corresponding to the probability distribution at each time-step
states: a list of decoder_size tensors, each corresponds to the internal state of the decoder at every time-step

Seq2seq in TensorFlow

```
outputs, states = embedding_rnn_seq2seq(encoder_inputs,
                                        decoder_inputs,
                                        cell,
                                        num_encoder_symbols,
                                        num_decoder_symbols,
                                        embedding_size,
                                        output_projection=None,
                                        feed_previous=False)
```

To embed your inputs and outputs, need to specify the number of input and output tokens
Feed_previous if you want to feed the previously predicted word to train, even if the model makes mistakes
Output_projection: tuple of project weight and bias if use sampled softmax

Seq2seq in TensorFlow

```
outputs, states = embedding_attention_seq2seq(encoder_inputs,
                                              decoder_inputs,
                                              cell,
                                               num_encoder_symbols,
                                               num_decoder_symbols,
                                               num_heads=1,
                                               output_projection=None,
                                               feed_previous=False,
                                              initial_state_attention=False)
```

Embedding sequence-to-sequence model with attention.

Wrapper for seq2seq with buckets

```
outputs, losses = model_with_buckets(encoder_inputs,
                                     decoder_inputs,
                                     targets,
                                     weights,
                                     buckets,
                                     seq2seq,
                                     softmax_loss_function=None,
                                     per_example_loss=False)
```

Seq2seq: one of the seq2seq functions defined above

Softmax loss function: normal softmax or sampled softmax

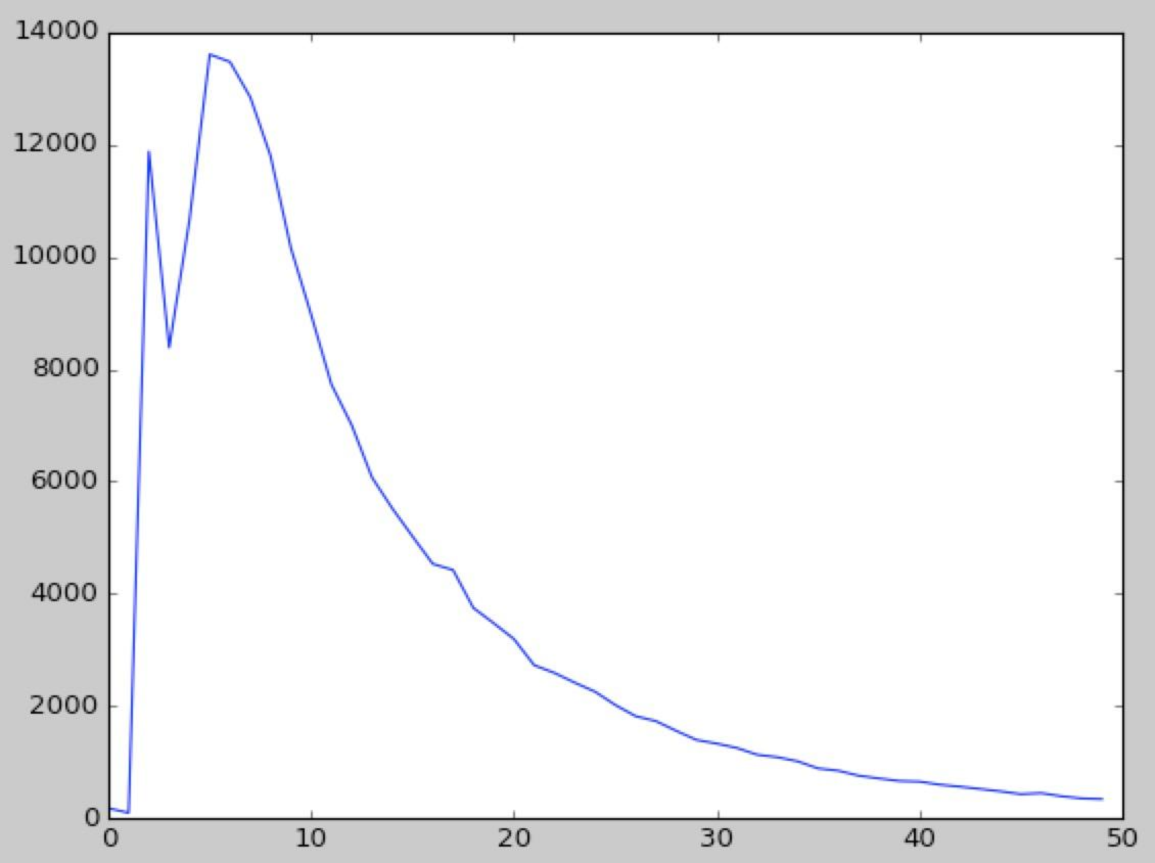
Our TensorFlow chatbot

Cornell Movie-Dialogs Corpus

- **220,579** conversational exchanges between
- **10,292** pairs of movie characters
- **9,035** characters from 617 movies
- **304,713** total utterances
- Very well-formatted (almost perfect)

Come with a very interesting paper "Chameleons in Imagined Conversations."

Input Length Distribution



Bucketing

9 buckets
[(6, 8), (8, 10), (10, 12), (13, 15), (16, 19), (19, 22), (23, 26), (29, 32), (39, 44)]
[19530, 17449, 17585, 23444, 22884, 16435, 17085, 18291, 18931]

5 buckets
[(8, 10), (12, 14), (16, 19), (23, 26), (39, 43)] # bucket boundaries
[37049, 33519, 30223, 33513, 37371] # number of samples in each bucket

3 buckets - recommended
[37899, 34480, 31045]
[(8, 10), (12, 14), (16, 19)]

Vocabulary tradeoff

- Get all tokens that appear at least a number of time (twice)
- Alternative approach: get a fixed size vocabulary

Smaller vocabulary:

- Has smaller loss/perplexity but loss/perplexity isn’t everything
- Gives <unk> answers to questions that require personal information
- Doesn’t give the bot’s answers much response
- Doesn’t train much faster than big vocab using sampled softmax

Model

- Seq2seq
- Attentional decoder
- Reverse encoder inputs
- Bucketing
- Sampled softmax
- Based on the Google's vanilla translate model, originally used to translate from English to French

The sassiest bot

```
> hi
you have you go .
> where?
in mind .
> you're weird?
you ' re right . . .
> what do you want?
who is you me .
> i don't understand
you know you look .
> i look what?
you nothing .
> you're funny
you ' re not .
```

> human
bot

The sassiest bot

```
> hi
hi .
> how're you?
just go .
> go where?
<unk> .
> omg
<unk> !
> you're doing that again!
i ' m not .
> stop telling me to go somewhere!
go !
> it's almost like you want to piss me off.
i ' m not .
> you're stupid
i ' m not .
```

> human
bot

It sort of works

```
> you are a monster!
i am . . .
> i'm sorry
you don ' t need to boy .
> i'm a girl!
you done ?
==========
> Did you change your hair?
no .
> no?
no i won t .
> who?
paul owen .
> where?
in a harem .
```

> human
bot

Sort of doesn’t

```
> what?
don ' t bother .
> you're being sassy
i need .
> what do you need?
i need ya help ya .
> okay
what do you want ?
```

> human
bot

Sort of doesn’t

```
> you're just talking gibberish
and what are you talking about ?
> about life
and what about it ?
> life is beautiful
is good .
> where do you live?
up .
> as in heaven?
in his apartment .
> who's he?
who ' s here ?
> oh well
and you are in trouble .
> i'm not!
please !
> whatever
i don ' t know what to say .
```

> human
bot

See sample conversations on GitHub (in assignments)

Sanity check?

How do we know that we implemented our model correctly?

Sanity check?

Run the model on a small dataset (~2,000 pairs) and run for a lot of epochs to see if it converges (learns all the responses by heart)

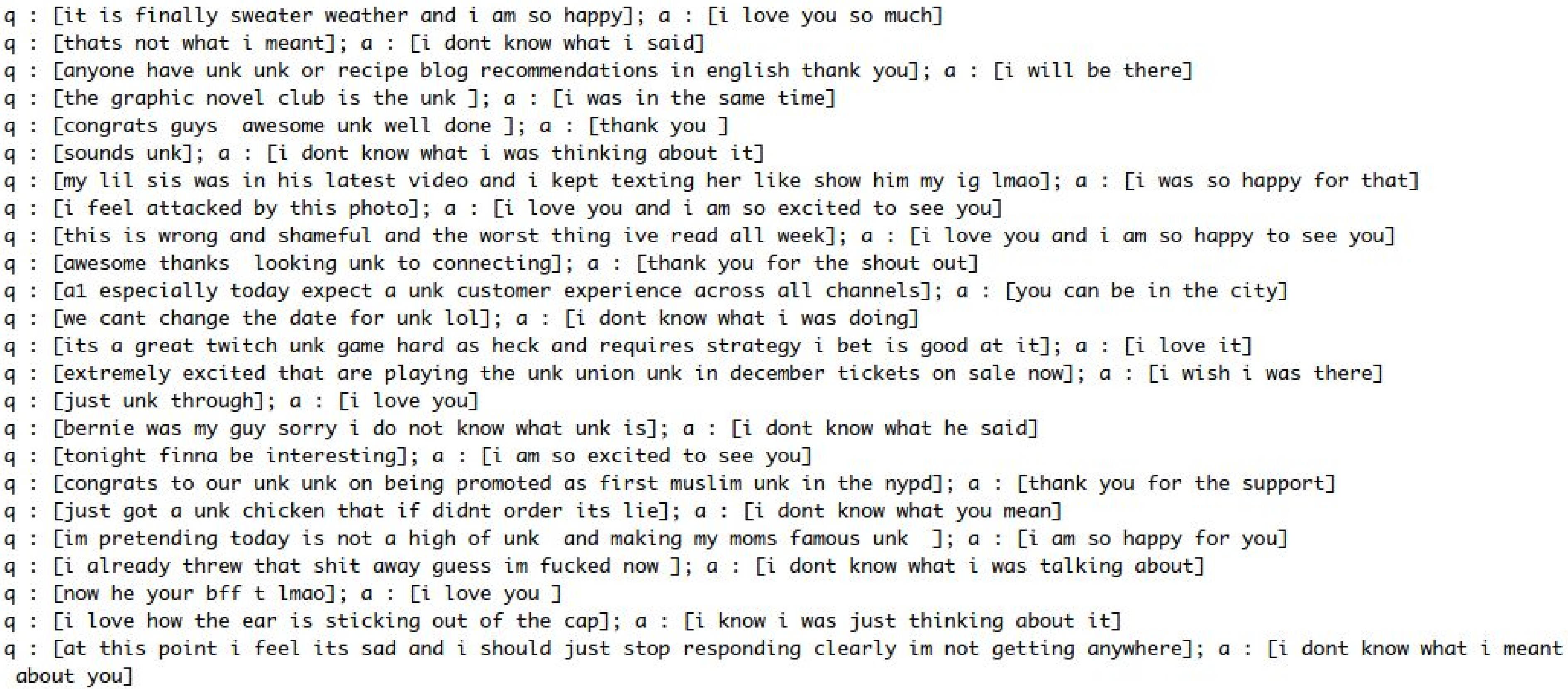
Problems?

- The bot is very dramatic (thanks to Hollywood screenwriters)
- Topics of conversations aren’t realistic
- Responses are always fixed for one encoder input
- Inconsistent personality
- Use only the last previous utterance as the input for the encoder
- Doesn’t keep track of information about users

Train on multiple datasets

- Twitter chat log (courtesy of Marsan Ma)
- More movie substitles (less clean)
- Every publicly available Reddit comments (1TB of data!)
- Your own conversations (chat logs, text messages, emails)

Example of Twitter chat log



```
q : [it is finally sweater weather and i am so happy]; a : [i love you so much]
q : [thats not what i meant]; a : [i dont know what i said]
q : [anyone have unk unk or recipe blog recommendations in english thank you]; a : [i will be there]
q : [the graphic novel club is the unk ]; a : [i was in the same time]
q : [congrats guys  awesome unk well done ]; a : [thank you ]
q : [sounds unk]; a : [i dont know what i was thinking about it]
q : [my lil sis was in his latest video and i kept texting her like show him my ig lmao]; a : [i was so happy for that]
q : [i feel attacked by this photo]; a : [i love you and i am so excited to see you]
q : [this is wrong and shameful and the worst thing ive read all week]; a : [i love you and i am so happy to see you]
q : [awesome thanks  looking unk to connecting]; a : [thank you for the shout out]
q : [a1 especially today expect a unk customer experience across all channels]; a : [you can be in the city]
q : [we cant change the date for unk lol]; a : [i dont know what i was doing]
q : [its a great twitch unk game hard as heck and requires strategy i bet is good at it]; a : [i love it]
q : [extremely excited that are playing the unk union unk in december tickets on sale now]; a : [i wish i was there]
q : [just unk through]; a : [i love you]
q : [bernie was my guy sorry i do not know what unk is]; a : [i dont know what he said]
q : [tonight finna be interesting]; a : [i am so excited to see you]
q : [congrats to our unk unk on being promoted as first muslim unk in the nypd]; a : [thank you for the support]
q : [just got a unk chicken that if didnt order its lie]; a : [i dont know what you mean]
q : [im pretending today is not a high of unk  and making my moms famous unk  ]; a : [i am so happy for you]
q : [i already threw that shit away guess im fucked now ]; a : [i dont know what i was talking about]
q : [now he your bff t lmao]; a : [i love you ]
q : [i love how the ear is sticking out of the cap]; a : [i know i was just thinking about it]
q : [at this point i feel its sad and i should just stop responding clearly im not getting anywhere]; a : [i dont know what i meant
 about you]
```

Chatbot with personalities

- At the decoder phase, inject consistent information about the bot
  For example: name, age, hometown, current location, job
- Use the decoder inputs from one person only
  For example: your own Sheldon Cooper bot!

Train on the incoming inputs

- Save the conversation with users and train on those conversations
- Create a feedback loop so users can correct the bot’s responses

Remember what users say

- The bot can extract information the user gives them

```
> hi
hi . what ' s your name ?
> my name is chip
nice to meet you .
> what's my name?
let ' s talk about something else .
```

Use characters instead of tokens

- Character level language modeling seems to be working quite well
- Smaller vocabulary -- no unknown tokens!
- But the sequences will be much longer (approximately 4x longer)

Improve input pipeline

- Right now, 50% of running time is spent on generating batches!

See assignment 3 handout

Next class

More discussion on chatbot

Feedback: huyenn@stanford.edu

Thanks!