1. Character-based convolutional encoder for NMT

(a)

Vocabulary size for characters is typically less than 100, which is 2 or even 3 magnitudes less than vocabulary size for works, that's why, meanwhile dense vectors of this 50 can hold much less information, than dense vectors of size 256, there is no need for capacity to be that big.

(b)

Character-based embedding model size:

$V_{char} * e_{char}$ parameters for embeddings.

$e_{word} * e_{char} * k + e_{word}$ parameters for convolutional layer.

$2 * e_{word} * e_{word} + 2 * e_{word}$ parameters for highway layer.

In total: $V_{char} * e_{char} + e_{word} * e_{char} * k + e_{word} + 2 * e_{word} * e_{word} + 2 * e_{word}$.

Word-based lookup embedding model size:

In total: $V_{word} * e_{word}$

Lets say that $e_{word} = 256$, $e_{char} = 50$, $k = 5$, $V_{word} = 50000$, $V_{char} = 96$ and compare models sizes:

Character-based embedding model size $= 96 * 50 + 256 * 50 * 5 + 256 + 2 * 256 * 256 + 2 * 256 = 200640$

Word-based lookup embedding model size $= 50000 * 256 = 12800000$

So, word-based lookup embedding model has more parameters, in 67 times.

(c)

As an output, convnet has an vector of dimension f, which is arbitrary and doesn't depend on e_{char}, by contrast, a RNN would has output of shape either e_{char} or $2 * e_{char}$, depending whether we use a bidirectional model or not, but the output size cannot be arbitrary. The convnet model, unlike a RNN, doesn't have such restriction.

(d)

Max-pooling has an advantage of capturing the information whether or not there was part of the sentence that fires the neuron, by contrast, average-pooling looks at all words in the sentence, including articles and other words that doesn't have much meaning.

Average-pooling has an advantage of capturing information the whole sentences, two completely different sentences can have same max-word and will be represented the same by max-pooling, by contrast, average-pooling will distinguish this sentences and contain more information about them.

(h)

Tests:

I create a simple model with word embedding size equals to 2, then check shapes of the parameters:

```
assert(model.proj.weight.shape == torch.Size([2, 2]))
assert(model.gate.weight.shape == torch.Size([2, 2]))
assert(model.proj.bias.shape == torch.Size([2]))
assert(model.gate.bias.shape == torch.Size([2]))
```

Then, create an input with batch_size = 4 and check that model predictions have right type, size and values:

```
with torch.no_grad():
    pred = model(x)
    assert(type(pred) == torch.Tensor)
    assert(pred.shape == torch.Size([4, 2]))
    assert(np.allclose(pred.numpy(), output))
```

(i) Tests:

I create a model with arbitrary number of in_channels, out_channels and kernel_size, 3, 2 and respectively. Then I check that parameters have the right shape:

```
assert(model.conv1d.weight.shape == torch.Size([2, 3, 4]))
assert(model.conv1d.bias.shape == torch.Size([2]))
```

Then, I create an input of batch_size = 2 and length = 5 and check the output of the model for the desired type and shape:

```
with torch.no_grad():
    pred = model(input)
    assert(type(pred) == torch.Tensor)
    assert(pred.shape == torch.Size([2, 2]))
    assert(np.allclose(pred.numpy(), output))
```

I believe that this test is sufficient, because by applying 2 convolutions to all positions in both examples in the batch obtained result is a combitation of negative-negative, negative-positive, positive-negative and positive-positive values:
kernel_1, batch_1: -2.3081 -1.7288
kernel_2, batch_1: 3.3081 2.7288
kernel_1, batch_2: 1.374 -0.987
kernel_2, batch_2: -0.37401 1.987
So the composition of ReLU and max-pooling is tested on all possible variants.

3. Analyzing NMT Systems

(a)

Occurs: traducir and traduce.

Doesn't occur: traduzco, traduces, traduzca, traduzcas.

Word-based NMT model wont be able to work properly with 4 of 6 of this word either changing them to <unk>s or coping withou translation, both technique are pretty bad.

Character-aware NMT model will be able to produce meaningful embeddings for words that is not in vocabulary because all of these words have the same prefix, so convolutions will be mostly over the same parts and will have the same output, for instance, traduce that occurs in the vocabulary and traduces that doesn't differs only in one character, and there embeddings will be really close as it supposed to be.

(b)

i.

financial - economic
neuron - nerve
Francisco - san
naturally - occurring
expectation - norms

ii.

financial - vertical
neuron - Newton
Francisco - France
naturally - practically
expectation - exception

iii.

Word2Vec models semantic similarities, it is clear from words that are close, for example, to financial - economic, business and markets.

CharCNN models characteristical similarities, lets have a look at words that are close to naturally - practically, typically, significantly, mentally and so on, all of them contains 'lly' part, same story with expectation - exception, indication, integration and so on, all words contains 'tion'.

Methodology in CharCNN basically applies learned kernels to different parts on words, and some kernels may correspond to 'lly' and 'tion' which means these parts activates kernels and then this activated value is being captured by max-pooling layer and resulting embeddings of naturally and practically becomes close. In contrast, Word2Vec defines embeddings according to the context of the word, not in parts, and since similar words appear in similar context financial and economic are close in embedding space.

(c)

Example of acceptable translation:

1. Tratamos de acabar con los encarcelamientos masivos".
2. We're trying to end mass incarceration."
3. We try to end the <unk> <unk>
4. We try to end up with mass infections."
5. Word «masivos"» with the quote at the end doesn't apper in vocab, but producing closest word to it, the one without quote, character-based decoder produces the right result.

Example of incorrect translation:

1. Tratamos de acabar con los encarcelamientos masivos".
2. We're trying to end mass incarceration."
3. We try to end the <unk> <unk>

4. We try to end up with mass infections."

5. Word «encarcelamientos» doesn't appear it vocabulary either, character-based decoder tries to produce something similar in the char-embedding space, but the meaning is totally different.