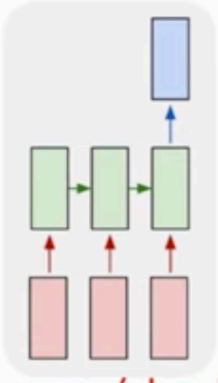
Train an RNN for Text Prediction

predict the next char



sequence (char-level)

input text

How do we train such an RNN?

- Cut text to segments (with overlap).
  - E.g., `seg_len=40` and `stride=3`.

Machine learning is a subset of artificial intelligence in the field of computer science that often uses statistical techniques to give computers the ability to "learn" (i.e., progressively improve performance on a specific task) with data, without being explicitly programmed. The name machine learning was coined in 1959 by Arthur Samuel.
…
…

1. Prepare Training Data

segment:

'preface\n\n\nsupposing that truth is a woman--what then? is there not ground\nfor suspecting that all philosophers, in so far as they have been\ndogmatists, have failed to understand women--that the te rrible\nseriousness and clumsy importunity with which they have usu ally paid\ntheir addresses to truth, have been unskilled and unseem

segments[0]: preface\n\n\nsupposing that truth **next_chars[0]:** i

segments[1]: face\n\n\nsupposing that truth is **next_chars[1]:** a

1. Prepare Training Data

segment:

```
'preface\n\n\nsupposing that truth is a woman--what then? is there
not ground\nfor suspecting that all philosophers, in so far as they
have been\ndogmatists, have failed to understand women--that the te
rrible\nseriousness and clumsy importunity with which they have usu
ally paid\ntheir addresses to truth, have been unskilled and unseem
```

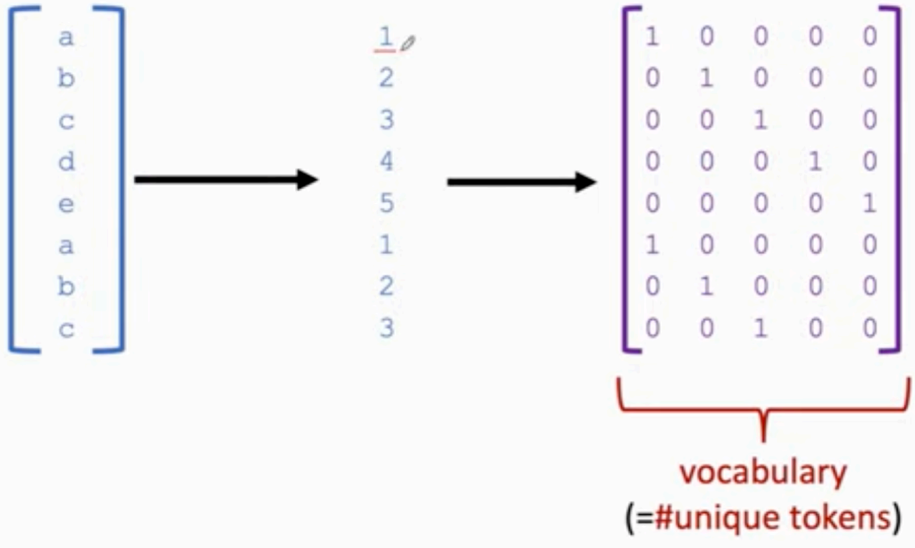
```
segments[0]: preface\n\n\nsupposing that truth    next_chars[0]: i
segments[1]: face\n\n\nsupposing that truth is    next_chars[1]: a
segments[2]: e\n\n\nsupposing that truth is a w   next_chars[2]: o
segments[3]: \nsupposing that truth is a woma     next_chars[3]: n
```

2. Character to Vector

Dictionary

Token	Index
a	1
b	2
c	3
d	4
e	5

One-hot encoding (token to vector)



2. Character to Vector

- Vocabulary is $v = 57$ (including letter, blank, punctuation, etc.)
- Segment length is $l = 60$. (A segment has 60 chars.)
- Number of segments is $n = 200{,}278$. (Number of training samples.)

`segments[i]:` `\nsupposing that truth is a woma` → **X**: 60×57 matrix

`next_chars[i]:` `n` → **y**: 57×1 vector

There are $n = 200{,}278$ such pairs.

3. Build a Neural Network

```
from keras import layers

model = keras.models.Sequential()
model.add(layers.LSTM(128, input_shape=(seg_len, vocabulary)))
model.add(layers.Dense(vocabulary, activation='softmax'))
model.summary()
```

$l = 60$ $v = 57$ $v = 57$

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 128)               95232
_________________________________________________________________
dense_1 (Dense)              (None, 57)                7353
=================================================================
Total params: 102,585
Trainable params: 102,585
Non-trainable params: 0
```

4. Train the Neural Network

```
optimizer = keras.optimizers.RMSprop(lr=0.01)
model.compile(loss='categorical_crossentropy', optimizer=optimizer)
```

```
model.fit(x, y, batch_size=128, epochs=1)
```

```
Epoch 1/1
200278/200278 [==============================] - 117s 586us/step -
loss: 1.6746
```

- **x[i,:,:]** is a segment of 60 chars represented by a 60×57 matrix.
- **y[i,:]** is the next char represented by a 57-dim vector.

Predict the Next Char

```
pred = model.predict(x_input, verbose=0)[0]
```

Question: Given the distribution, what will be the next char?

- Option 1: greedy selection.
  - `next_index = np.argmax(pred)`
  - It is deterministic.
  - Empirically not good.

Predict the Next Char

```
pred = model.predict(x_input, verbose=0)[0]
```

Question: Given the distribution, what will be the next char?

- Option 1: greedy selection.
- Option 2: sampling from the multinomial distribution.
  - `next_onehot = np.random.multinomial(1, pred, 1)`
  - `next_index = np.argmax(next_onehot)`
  - Maybe too random.

Predict the Next Char

```
pred = model.predict(x_input, verbose=0)[0]
```

Question: Given the distribution, what will be the next char?

- Option 1: greedy selection.
- Option 2: sampling from the multinomial distribution.
- Option 3: adjusting the multinomial distribution.
  - `pred = pred ** (1 / temperature)`
  - `pred = pred / np.sum(pred)`
  - Sample according to `pred`.
  - Between greedy and multinomial (controlled `temperature`).

Predict the Next Char

Option 3: adjusting the multinomial distribution.

- `pred = pred ** (1 / temperature)`
- `pred = pred / np.sum(pred)`
- Between greedy and multinomial (controlled `temperature`).

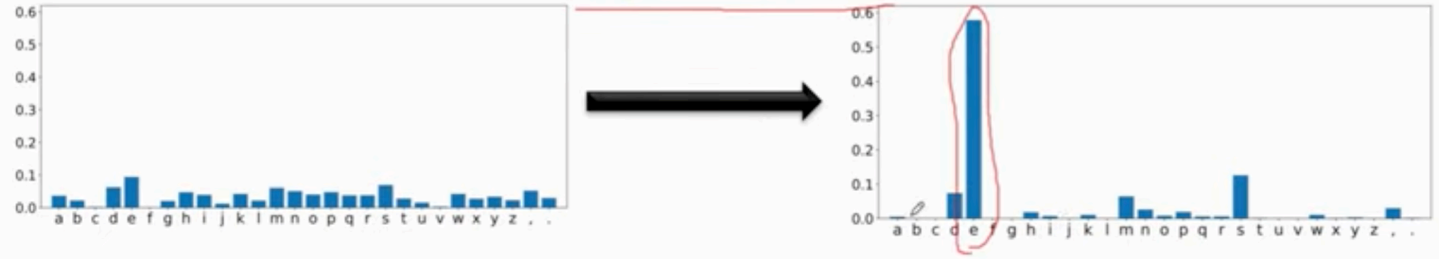
`temperature=0.5`

Predict the Next Char

Option 3: adjusting the multinomial distribution.

- `pred = pred ** (1 / temperature)`
- `pred = pred / np.sum(pred)`
- Between greedy and multinomial (controlled `temperature`).

`temperature=0.2`



Text Generation: An Example

Example: `seg_len=18`

initial_input (seed): "the cat sat on the"

next_char: '_' (blank)

input: "he cat sat on the "

next_char: 'm'

input: "e cat sat on the m"

Train a Neural Network

1. Partition text to (segment, next_char) pairs.
2. One-hot encode the characters.
   - Character → $v \times 1$ vector.
   - Segment → $l \times v$ matrix.
3. Build and train a neural network.
   - $l \times v$ matrix ==> LSTM ==> Dense ==> $v \times 1$ vector.

Text Generation

1. Propose a seed segment.
2. Repeat the followings:
   a) Feed the segment (with one-hot) to the neural network.
   b) The neural network outputs probabilities.
   c) next_char ← Sample from the probabilities.
   d) Append next_char to the segment.