# Natural Language Processing with Entropy

DSTA

### 0.1 NLP and Entropy measures

### 0.2 Motivations

Today, Natural language processing (NLP) is advancing thanks (mostly) to the deployment of sophisticated Machine Learning architectures.

A key enabling factor is the ready availability of large corpora which feed extensive training of the ML architectures

---

New computational problems are now addressed. Example: given a prompt text, complete the phrase in an intelligible (i.e., human) way.

. . .

Entropy helps us understand and evaluate progress in NL, e.g., in language generation.

### 0.3 Agenda

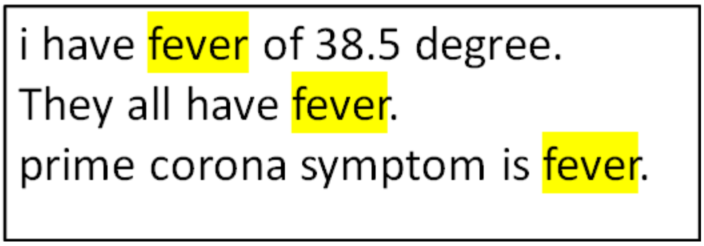
- A general introduction to NLP
- Language models: a quick glance
- introduction of Cross-entropy

# 1 Current Topics in NLP

## 1.1 Named entities

Suppose, we want to extract Coronavirus symptoms from a stream of Twitter posts dedicated to Covid-19.

Finding a new symptom is the task of *Named Entity Recognition (NER).*




---

Definition:

given a text find all persons, locations and organisations that the text refers to.

## 1.2 More challenges

### 1.2.1 Language generation (prompt generation)

given the initial part of a phrase, called `prompt` give a *proper* completion of the phrase.

. . .

#### 1.2.1.1 Question answering

Given a question in natural language, reply to it *properly.*

We need a metrics to determine what is *proper* for a solution.

---

#### 1.2.1.2 Phrase completion

**Instance:** a phrase/sequence of words W

**Solution:** the next word(s) in the phrase.

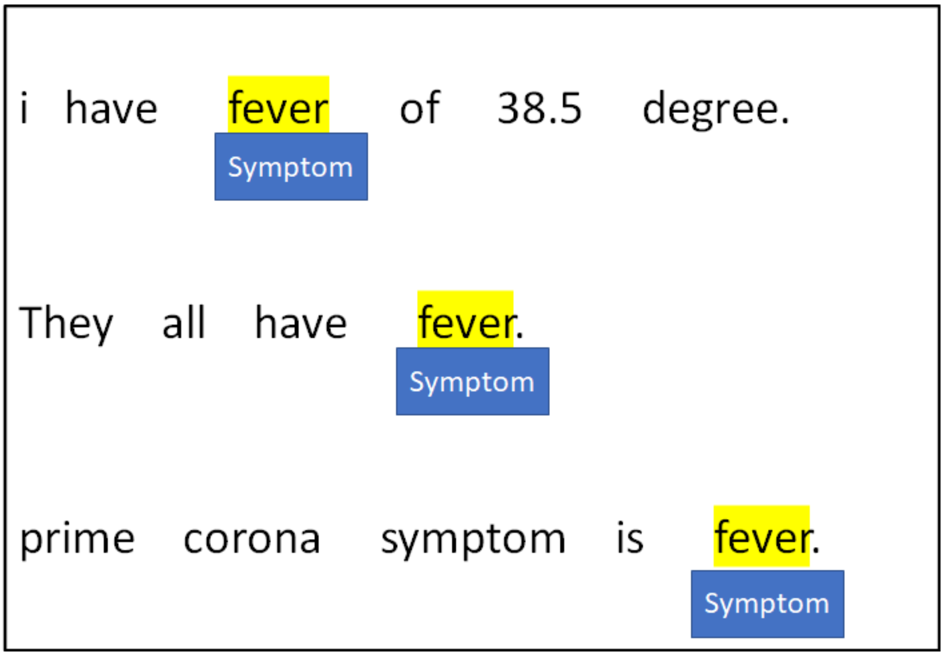
. . .

We will focus on this problem.

## 1.3 Supervised ML

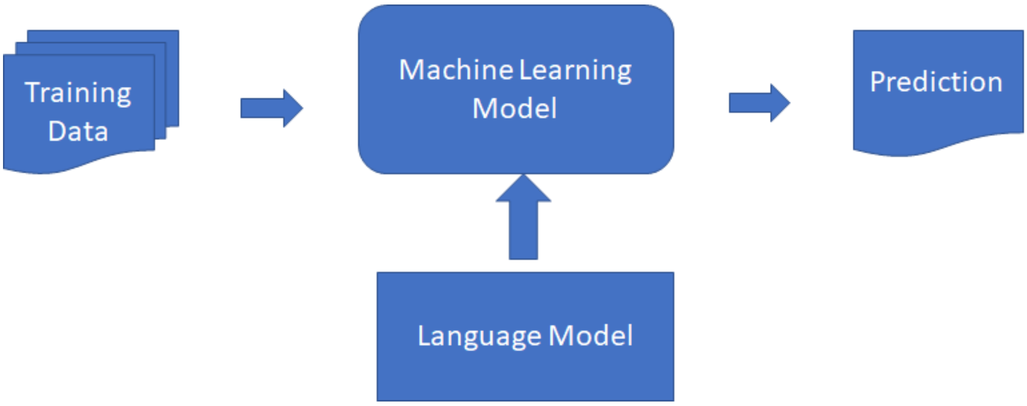
Named entity recognition (and others) are normally addressed by means of a *supervised Machine Learning model.*

It starts with human, reliable annotation of example texts: the training data.




---

If *well-trained*, the ML model will be capable of predicting the entities in new text.



## 2 Phrase Completion

### 2.1 Predicting the next word

An algorithm which assigns probabilities to sequences of words is called a *language model (LM).*

The simplest model assigns probabilities to sentences and sequences of words, the *n-gram.*

### 2.2 N-gram models

An n-gram is a sequence of n words:

2-gram (bigram): “switch off” or ”your homework”

3-gram (trigram): “please turn your,” or “turn your homework”.

---

N-gram prediction is based on probabilities.
Probs. are extracted from frequencies of co-occurence in corpora.

. . .

This bigram tabled by [Martin-Jufrasky] is a LM in itself

| | i | want | to | eat | chinese | food | lunch | spend |
|---|---|---|---|---|---|---|---|---|
| **i** | 5 | 827 | 0 | 9 | 0 | 0 | 0 | 2 |
| **want** | 2 | 0 | 608 | 1 | 6 | 6 | 5 | 1 |
| **to** | 2 | 0 | 4 | 686 | 2 | 0 | 6 | 211 |
| **eat** | 0 | 0 | 2 | 0 | 16 | 2 | 42 | 0 |
| **chinese** | 1 | 0 | 0 | 0 | 0 | 82 | 1 | 0 |
| **food** | 15 | 0 | 15 | 0 | 1 | 4 | 0 | 0 |
| **lunch** | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| **spend** | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

**Figure 3.1** Bigram counts for eight of the words (out of $V = 1446$) in the Berkeley Restaurant Project corpus of 9332 sentences. Zero counts are in gray.

### 2.3 How to evaluate results

N-gram models estimate the probability of the last word of an n-gram given the previous words.

| | i | want | to | eat | chinese | food | lunch | spend |
|---|---|---|---|---|---|---|---|---|
| **i** | 5 | 827 | 0 | 9 | 0 | 0 | 0 | 2 |
| **want** | 2 | 0 | 608 | 1 | 6 | 6 | 5 | 1 |
| **to** | 2 | 0 | 4 | 686 | 2 | 0 | 6 | 211 |
| **eat** | 0 | 0 | 2 | 0 | 16 | 2 | 42 | 0 |
| **chinese** | 1 | 0 | 0 | 0 | 0 | 82 | 1 | 0 |
| **food** | 15 | 0 | 15 | 0 | 1 | 4 | 0 | 0 |
| **lunch** | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| **spend** | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

**Figure 3.1** Bigram counts for eight of the words (out of $V = 1446$) in the Berkeley Restaurant Project corpus of 9332 sentences. Zero counts are in gray.

Which is the best N-gram model?

To rate them we need Information theory.

# 3 Cross Entropy

---

### 3.1 Approach

Let's consider N-grams.

What is the best N-gram model for predicting the next word of a sequence W?

Information theory provides an abstract yet effective tool to evaluate the quality of solutions (against some test data).

### 3.2 Preliminaries

Let P(i) and Q(i) be two prob. distributions drawn from the same underlying set of $n$ possible *outcomes:* $X = \{x_1, \ldots x_n\}$.

Let P(i) be the current distribution: the text to be evaluated.

Let Q(i) be the *reference distribution,* given by the model (trained on the whole corpus).

The cross-entropy $H(P, Q)$ is defined as

$$H(P,Q) = -\sum_{i=1}^{n} p(x_i) \log q(x_i)$$

### 3.3 Perplexity

Cross-entropy: $H(P,Q) = -\sum_{i=1}^{n} p(x_i) \log q(x_i)$

. . .

Perplexity: $PP(H,Q) = 2^{H(P,Q)}$.

A lower perplexity indicates a better model.

### 3.4 Computing perplexity

Large-scale NLP models provide a convenient approximation

Let $W = w_1, \ldots, w_N$ be the phrase at hand.

For each word occurence $w_i$, the model assigns a prob. $P(w_i)$.

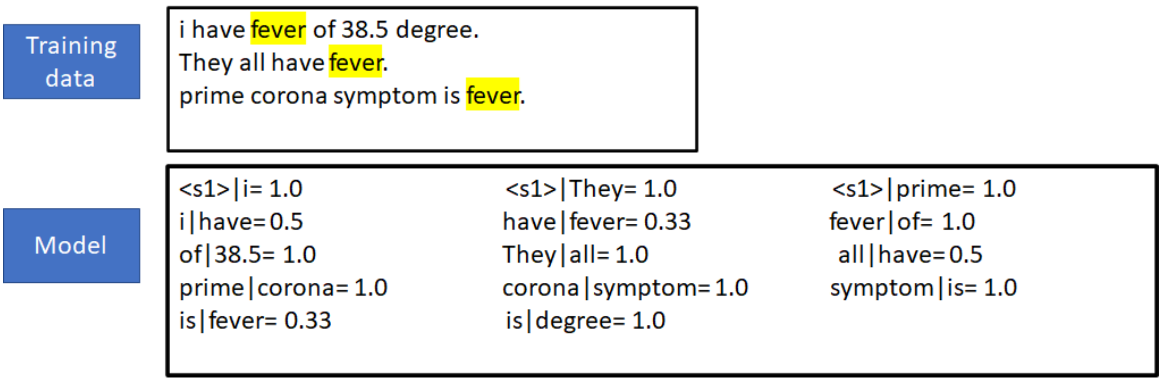
Now, Entropy approximates the cross-entropy of W:

$H(W) \approx -\frac{1}{N} \log P(W)$

. . .

$H(W) \approx -\frac{1}{N} \log \Pi_{i=1}^{N} P(w_i)$

### 3.5 Perplexity in action



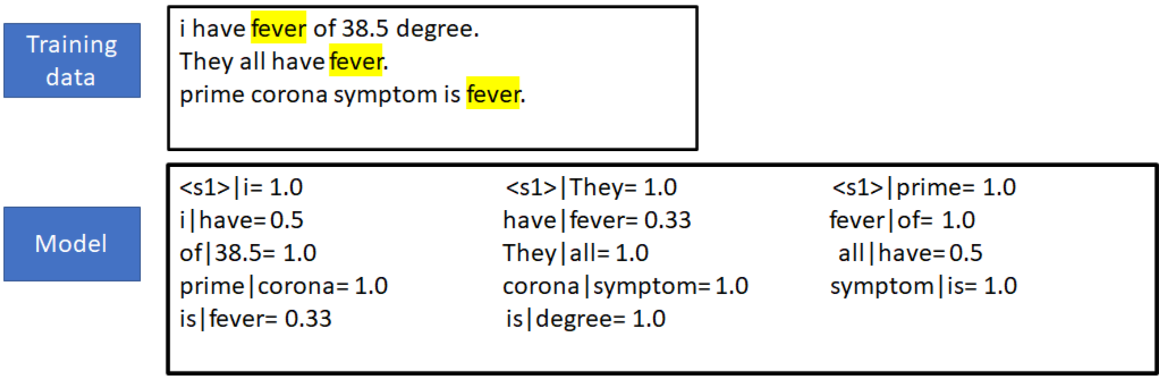

What is the perplexity of the following test sentence (notice grammar)?

W = `prime corona symptom is fever and cough`

---

W $=$ `prime corona symptom is fever and cough`

W is now our P distribution




$\frac{1}{N} = \frac{1}{7}$

$P(W) = \Pi_{i=1}^{N} P(w_i) = 1 \times 1 \times 1 \times 1 \times 0.33 \times 1 \times 1 = 0.33$

---

Let's compute the Cross entropy of W:

$H(W) \approx -\frac{1}{N} \log \Pi_{i=1}^{N} P(w_i)$

. . .

$H(W) = -\frac{1}{7} \cdot \log(1 \cdot 1 \cdot 1 \cdot 1 \cdot 0.33 \cdot 1 \cdot 1)$

. . .

$H(W) = -\frac{1}{7} \cdot (-0.625) = 0.089$

$PP(W) = 2^{H(W)} = 2^{0.089} = 1.063$

### 3.6 Mastering perplexity

$H(W)$ is the average number of bits needed to encode each word.

$P(W) = 2^{H(W)}$ is the average number of words that can be encoded using H(W) bits.

. . .

**However. . .**

---

We interpret perplexity as a weighted branching factor for the possible completions of W.

If PP=1 there is no doubt on what the next word should be.

If PP=100 then whenever the model is trying to guess the next word it is as confused as if it had to pick between 100 words.

---

W = `I study at Birkbeck...`

which is the next likely word?

. . .

$M_1$: `College`

with perplexity 3: `University` and `London` are also highly probable completions.

$M_2$: `University`

with perplexity 1.5: `College` is the only medium-probability alternative.

We prefer $M_2$ as the lowest-perplexity model.