Probability power: language and topic modelling

Stanislav Protasov

Agenda

- First, mostly of **language modelling**
- And then, little on **topic modelling**

Language Model: beginning

Language model is used to *generate* texts (e.g. suggest next word) or to compare texts being similar in some case ("*recognize*")

Prerequisites

What is mathematical **model**?

Why do we need **model**? (in general)

Which **models** did we already consider, and how did we **benefit**:

- Vector Space Model
- Deep ANNs
- Small World Model
- Regular languages for syntax modelling
- ...

What is a language

Language is a set. **Language model** serves for two tasks:

- **Guarding predicate** `L = {w|P(w)}  # recognition`
- Set **generation procedure**

And as any other model, it helps to solve tasks stated for its prototype (real language):

- Are these words similar?
- What is the answer for the question?
- Are those texts from the same topics?
- ...
- **propose your own tasks**

Did we already cover any **language models**?

- Vector space and latent space model (*)
- Chomsky hierarchy models…
  - including finite state automata

Very simple model (from the book)

This is a sum …

$$\sum_{s \in \Sigma^*} P(s) = 1$$

This is a language set (all sentences)

$$P_{\text{uni}}(t_1 t_2 t_3) = P(t_1)P(t_2)P(t_3).$$

Unigram model

UM — token (word) frequency as probability estimation

$$P(s) =$$

$$= P(\text{“abbcccd”}) = P(\text{“abbcccd[STOP]”}) =$$

$$= P(\text{“[~S]a[~S]b[~S]b[~S]c[~S]c[~S]c[~S]d[STOP]”}) =$$

$$= [\ P(\text{“a”}) * P(\text{“b”})^2 * P(\text{“c”})^3 * P(\text{“d”})\ ] * [P(\sim STOP)^7 * P(STOP)]$$

Constant for sentences of equal length

Unigram model

$$P(\text{query}) = \prod_{\text{term in query}} P(\text{term})$$

Consider {a, b} as vocabulary.

P($) = ½

P(a$) =P(b$) = ½ * ½^2 = ⅛ | ¼

P(aa$) =P(ba$) = = ½ * ½ * ½ ^{3} = 1/32 | ⅛

In practice **constant part** is omitted:

1. For same length this is constant
2. In search we compare models. For model comparison constant factor is omitted

Model types

General model $$P(t_1 t_2 t_3 t_4) = P(t_1)P(t_2|t_1)P(t_3|t_1 t_2)P(t_4|t_1 t_2 t_3)$$

Unigram model $$P_{\text{uni}}(t_1 t_2 t_3 t_4) = P(t_1)P(t_2)P(t_3)P(t_4)$$

Bigram model $$P_{\text{bi}}(t_1 t_2 t_3 t_4) = P(t_1)P(t_2|t_1)P(t_3|t_2)P(t_4|t_3)$$

How to find $P(t_1)$, $P(t_2|t_1)$?

Unigram model under the microscope == multinomial distribution

We don’t care about the order, that is why sentence is a bag of words:

```
Professor called Stas ate a dog == Dog called Stas ate a professor
```

Probability of bag ***d*** then is described by a multinomial distribution:

$$P(d) = \frac{L_d!}{\mathrm{tf}_{t_1,d}!\mathrm{tf}_{t_2,d}! \cdots \mathrm{tf}_{t_M,d}!} P(t_1)^{\mathrm{tf}_{t_1,d}} P(t_2)^{\mathrm{tf}_{t_2,d}} \cdots P(t_M)^{\mathrm{tf}_{t_M,d}}$$

Here, $L_d = \sum_{1 \le i \le M} \mathrm{tf}_{t_i,d}$ is the length of document d, M is the size of the term vocabulary, and the products are now over the terms in the vocabulary, not the positions in the document.

Language Model: how to use it in IR

Language model is another way of **comparing texts.** For this you can use language models of query and text. And look for such text, which are “better” in generating queries.

Use **KL-divergence** for this.

Ranking, oh yeah!

Likelihood

$$L(\theta \mid x) = p_\theta(x) = P_\theta(X = x)$$

Query Likelihood Model $L(q_{\text{parameter}}|d_{\text{var}})$

$$P(d|q) = P(q|d)P(d)/P(q)$$

$$P(d \mid q) \sim P(q \mid d)$$

$$P(d) = \frac{L_d!}{\mathrm{tf}_{t_1,d}!\mathrm{tf}_{t_2,d}!\cdots\mathrm{tf}_{t_M,d}!} P(t_1)^{\mathrm{tf}_{t_1,d}} P(t_2)^{\mathrm{tf}_{t_2,d}} \cdots P(t_M)^{\mathrm{tf}_{t_M,d}}$$

Each text is a language! ლ(ಠ益ಠლ)

1. Build a model for each text (does it look similar to building TDM?)
2. Compute relevance for each doc (how **likely** this **query** is generated **by the doc language**)

$$P(q|M_d) = K_q \prod_{t \in V} P(t|M_d)^{\mathrm{tf}_{t,d}}$$

But wait, we have ***the*** language model M_c

1. If $\hat{P}(t|M_d) = 0$ ($\boldsymbol{tf_{t,d}} = 0$) $\hat{P}(t|M_d) \leq \mathrm{cf}_t / T$

2. Linear interpolation

$$\hat{P}(t|d) = \lambda \hat{P}_{\mathrm{mle}}(t|M_d) + (1-\lambda)\hat{P}_{\mathrm{mle}}(t|M_c)$$

3. Bayesian smoothing $\hat{P}(t|d) = \dfrac{\mathrm{tf}_{t,d} + \alpha \hat{P}(t|M_c)}{L_d + \alpha}$

Summary

1. Unigram model $P_{\text{uni}}(t_1 t_2 t_3 t_4) = P(t_1)P(t_2)P(t_3)P(t_4)$
2. Unigram max likelihood query $L(d|q) \sim P(q|d)$
3. $P(t|d)$!= $\text{tf}_{t,d}$ / D as it leads to 0-s
4. Estimate using **global** language **model**

$$P(d|q) \propto P(d) \prod_{t \in q} ((1-\lambda)P(t|M_c) + \lambda P(t|M_d))$$

Extended LM and motivations

Add **query language model** and use Kullback-Leibler divergence

$$R(d;q) = KL(M_d \| M_q) = \sum_{t \in V} P(t|M_q) \log \frac{P(t|M_q)}{P(t|M_d)}$$

What about **multilingual retrieval**? Just add translation matrix!

$$P(q|M_d) = \prod_{t \in q} \sum_{v \in V} P(v|M_d) T(t|v)$$

Topic modelling

There are (latent) topics, which define behaviour of language models. **Text belongs to a combination of few topics**, which can be discovered using TM.

Topic model

Purpose:

1. Which **topics** a **document** belongs to
2. Which **words** form which **topic**

Which questions answer:

1. How did the topic evolve in media?
2. What are major topics of this author?
3. How many topics are there?
4. ...

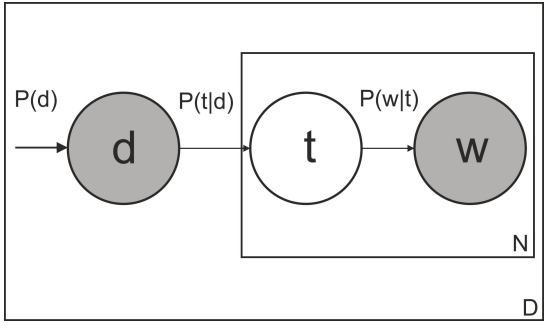
Conditional independence hypothesis

How **words are generated** in the model does not depend on ~~**particular document**~~, but **on the topic** of the document.

$$
\begin{aligned}
p(w \mid d, t) &= p(w \mid t); \\
p(d \mid w, t) &= p(d \mid t); \\
p(d, w \mid t) &= p(d \mid t) p(w \mid t).
\end{aligned}
$$

Topic model is (formally)

$$p(d,w) = \sum_{t \in T} p(t)p(w|t)p(d|t) = \sum_{t \in T} p(d)p(w|t)p(t|d) = \sum_{t \in T} p(w)p(t|w)p(d|t)$$



$$p(w \mid d) = \sum_{t \in T} p(t \mid d)\, p(w \mid t).$$

***p*(*w*|*d*)** - how to estimate and model these values?

$$\Phi = ||p(w|t)||$$

$$\Theta = ||p(t|d)||$$

Looking for decomposition F ≈ ΘΦ!

Sparseness hypothesis: document belongs to a very limited topic number. Thus, both Θ and Φ are sparse.

They model **probabilities**, thus, **p(*|*) ∈ [0, 1]** and

$$\sum_w p(w|t) = 1, \sum_t p(t|d) = 1, \sum_t p(t) = 1,$$

SVD and PCA don't work (non-stochastic matrix + bias)

Scientists argue whether **PLSA** (Vorontsov) or **LDA** (Blei, Ng) is better

B(y|e)!