Language models

Vlad Shakhuro

24 October 2025

Outline

1. What is Language Modelling?
2. N-Gram Language Models
3. Neural Network Language Models
4. Generation strategies
5. Evaluation
6. Examples

What is Language Modelling?

Language Modelling is the task of predicting what word comes next

the students opened their _____ (books, laptops, exams, minds)

More formally given context tokens $x_1, x_2, \dots, x_t$ compute probability distribution of the next token x_{t+1}:

$$P(x_{t+1} \mid x_1, \dots, x_t)$$

A system that solves this task is called a **Language Model**

Why care about Language Modelling?

- Language Modelling is a **benchmark task** to measure our progress on predicting language use
- Language Modelling is a **subcomponent** of many NLP tasks, especially those involving generating text or estimating the probability of text:
  - predictive typing
  - speech recognition
  - handwriting recognition
  - spelling/grammar correction
  - machine translation
  - summarization
- Everything else in NLP has been rebuilt upon Language Modelling

Outline

1. What is Language Modelling?
2. N-Gram Language Models
3. Neural Network Language Models
4. Generation strategies
5. Evaluation
6. Examples

Directly counting occurences

$$P(x_t \mid x_1, \ldots, x_{t-1}) = \frac{N(x_1, \ldots, x_t)}{N(x_1, \ldots, x_{t-1})}$$

Directly counting occurences

$$P(x_t \mid x_1, \dots, x_{t-1}) = \frac{N(x_1, \dots, x_t)}{N(x_1, \dots, x_{t-1})}$$

To make computation robust we use Markov assumption for n-gram model:

$$P(x_t \mid x_1, \dots, x_t) = P(x_t \mid x_{t-n+1}, \dots, x_{t-1})$$

For instance,

- $n = 3$: $P(x_t \mid x_1, \dots, x_{t-1}) = P(x_t \mid x_{t-2}, x_{t-1})$
- $n = 2$: $P(x_t \mid x_1, \dots, x_{t-1}) = P(x_t \mid x_{t-1})$
- $n = 1$: $P(x_t \mid x_1, \dots, x_{t-1}) = P(x_t)$

Backoff

$$P(\text{meaning} \mid \text{building blocks of}) = \frac{N(\text{building blocks of meaning})}{N(\text{building blocks of})}$$

Backoff

$$P(\text{meaning} \mid \text{building blocks of}) = \frac{N(\text{building blocks of meaning})}{N(\text{building blocks of})}$$

If count in denominator is zero, try shorter context:

$$P(\text{meaning} \mid \text{building blocks of}) \approx P(\text{meaning} \mid \text{blocks of})$$

or

$$P(\text{meaning} \mid \text{building blocks of}) \approx P(\text{meaning} \mid \text{of})$$

or

$$P(\text{meaning} \mid \text{building blocks of}) \approx P(\text{meaning})$$

Linear interpolation

$$P(\text{meaning} \mid \text{building blocks of}) \approx \lambda_0 P(\text{meaning} \mid \text{building blocks of}) + \lambda_1 P(\text{meaning} \mid \text{blocks of}) + \lambda_2 P(\text{meaning} \mid \text{of}) + \lambda_3 P(\text{meaning})$$

$$\sum_i \lambda_i = 1$$

Laplace smoothing

$$P(\text{meaning} \mid \text{building blocks of}) = \frac{N(\text{building blocks of meaning})}{N(\text{building blocks of})} \approx \frac{\delta + N(\text{building blocks of meaning})}{\delta \cdot |V| + N(\text{building blocks of})}$$

Pretend we saw each n-gram at least δ times

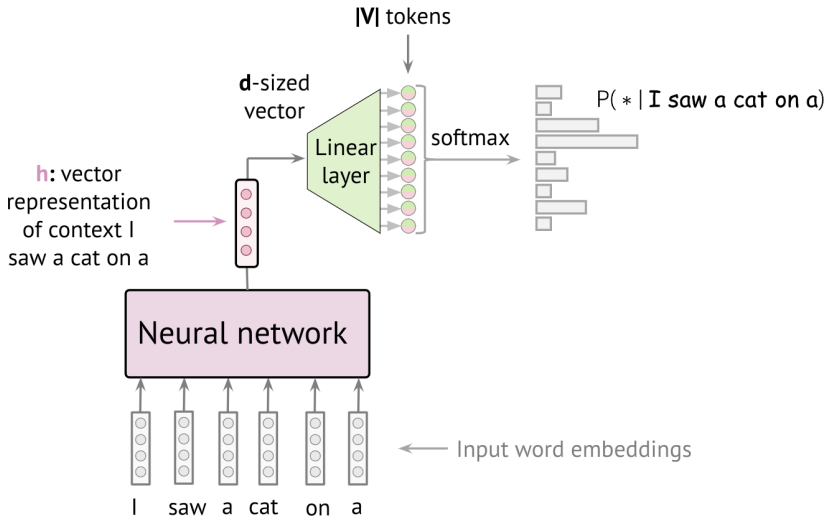
Generating text with n-gram model

```
when this option may be the worst day of amnesty
international delegations visited israel , and felt that
his sisters , that they are reserved for zyryanovsk
concentrating factory there is a member of the shire ,"
given as to damage the expansion of a meeting over a large
health maintenance organization , smoking ,
airconditioning , designated smoking area . _eos_
```

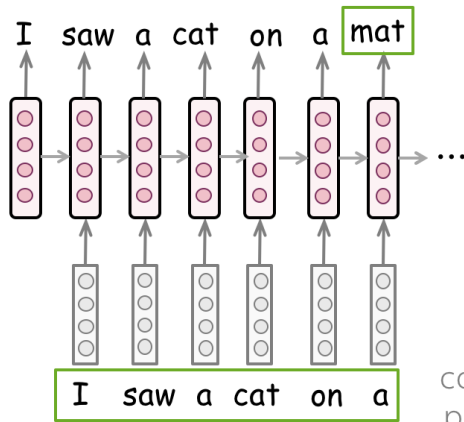
Outline

1. What is Language Modelling?
2. N-Gram Language Models
3. Neural Network Language Models
4. Generation strategies
5. Evaluation
6. Examples

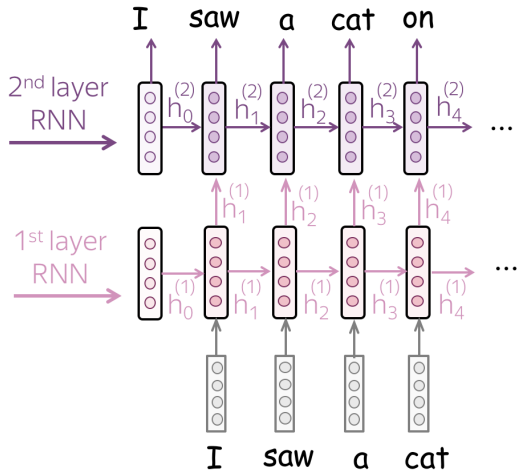
NN pipeline


|V| tokens
d-sized vector
Linear layer
softmax
P(* | I saw a cat on a)
h: vector representation of context I saw a cat on a
Neural network
Input word embeddings
I
saw
a
cat
on
a

NN pipeline


I saw a cat on a mat
...
I saw a cat on a
predict the next token
condition on the previous tokens

NN pipeline


I
saw
a
cat
on
2nd layer RNN
1st layer RNN
(2) h0
(2) h1
(2) h2
(2) h3
(2) h4
(1) h1
(1) h2
(1) h3
(1) h4
(1) h0
(1) h1
(1) h2
(1) h3
(1) h4
...
...
I
saw
a
cat

Outline

1. What is Language Modelling?
2. N-Gram Language Models
3. Neural Network Language Models
4. Generation strategies
5. Evaluation
6. Examples

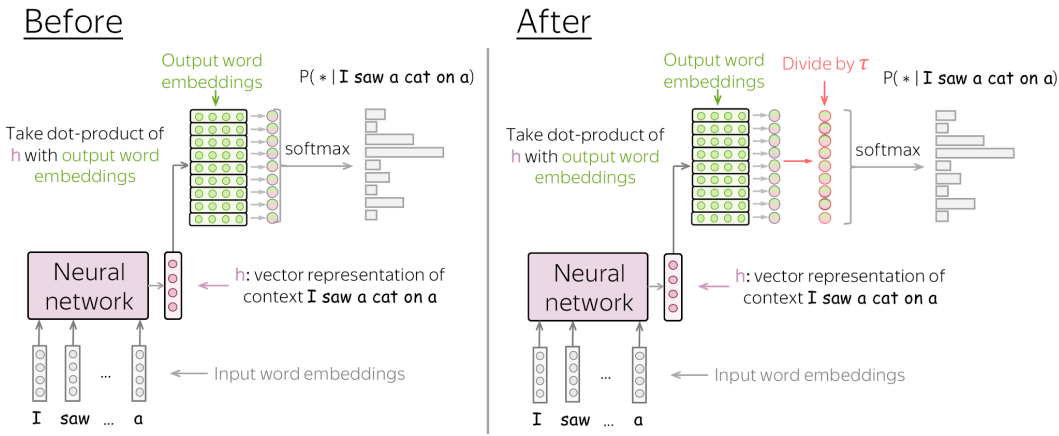
Diversity and coherence

Usually, we need texts to be:

- coherent — the generated text has to make sense,
- diversity — the model has to be able to produce very different samples.

Sampling with temperature

$$\frac{\exp(h^t w)}{\sum_{w_i \in V} \exp(h^t w_i)} \rightarrow \frac{\exp(\frac{h^t w}{\tau})}{\sum_{w_i \in V} \exp(\frac{h^t w_i}{\tau})}$$



High temperature

paradise sits farms started paint hollow almost unprecedented decisions, care using withdrawal from rebel cis (, saying graphics mongolia official line, greeted agenda victor is exploring anger :) draw testify liberalization decay productive 2 went exchanges of marketing drawing enabling challenging systematic crisis influencing the executive arrangement performs designs

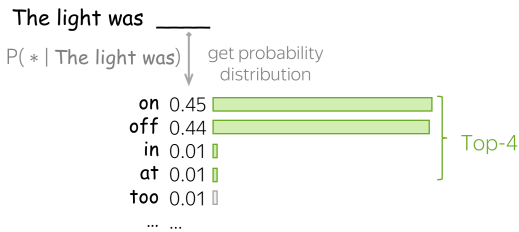
Low temperature

```
the first time the two - year - old - old girl with a
new version of the new version of the new version of
the new version of the new version of the new version
of the new version of the new version of the new
version of the
```

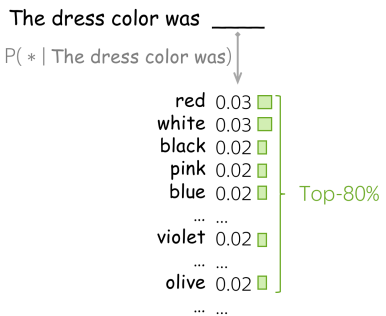
Top-K sampling

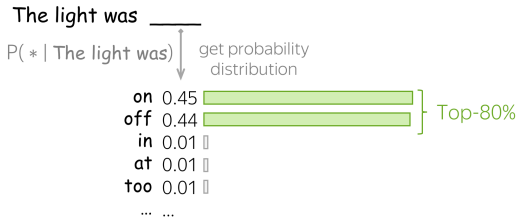
Top-K for a flat distribution: not enough

Top-K for a peaky distribution: too many



Top-p (nucleus) sampling





Outline

1. What is Language Modelling?
2. N-Gram Language Models
3. Neural Network Language Models
4. Generation strategies
5. Evaluation
6. Examples

Perplexity

Our loss (cross-entropy) is negative log-likelihood:

$$L(x_1, \ldots, x_M) = -\sum_{t=1}^{M} \log_2 P(x_t \mid x_{<t})$$

Its exponentiation is called perplexity:

$$\text{Perplexity}(x_1, \ldots, x_M) = 2^{\frac{L(x_1, \ldots, x_M)}{M}}$$

- The best perplexity is 1:

  If the model is perfect and assigns probability 1 to correct tokens, then the log-probabilities are zero

- The worst perplexity is $|V|$:

  If the model knows nothing about the data, it assigns probability $1/|V|$ to all tokens, regardless of context

Outline

1. What is Language Modelling?
2. N-Gram Language Models
3. Neural Network Language Models
4. Generation strategies
5. Evaluation
6. Examples

LaTeX generation

Proof. Omitted. □

Lemma 0.1. *Let $\mathcal{C}$ be a set of the construction.*

Let $\mathcal{C}$ be a gerber covering. Let $\mathcal{F}$ be a quasi-coherent sheaves of $\mathcal{O}$-modules. We have to show that

$$\mathcal{O}_{\mathcal{O}_X} = \mathcal{O}_X(\mathcal{L})$$

.

Proof. This is an algebraic space with the composition of sheaves $\mathcal{F}$ on $X_{\acute{e}tale}$ we have

$$\mathcal{O}_X(\mathcal{F}) = \{morph_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

where $\mathcal{G}$ defines an isomorphism $\mathcal{F} \to \mathcal{F}$ of $\mathcal{O}$-modules. □

Lemma 0.2. *This is an integer $\mathcal{Z}$ is injective.*

Proof. See Spaces, Lemma **??**. □

Lemma 0.3. *Let S be a scheme. Let X be a scheme and X is an affine open covering. Let $\mathcal{U} \subset \mathcal{X}$ be a canonical and locally of finite type. Let X be a scheme. Let X be a scheme which is equal to the formal complex.*

The following to the construction of the lemma follows.

Let X be a scheme. Let X be a scheme covering. Let

$$b : X \to Y' \to Y \to Y \to Y' \times_Y X \to X.$$

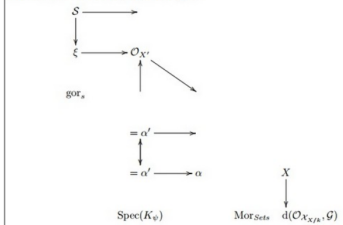
be a morphism of algebraic spaces over S and Y.

Proof. Let X be a nonzero scheme of X. Let X be an algebraic space. Let $\mathcal{F}$ be a quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent

(1) $\mathcal{F}$ is an algebraic space over S.
(2) If X is an affine open covering.

Consider a common structure on X and X the functor $\mathcal{O}_X(U)$ which is locally of finite type. □

This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram



is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type f_*. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

□

Proof. We have see that $X = \mathrm{Spec}(R)$ and $\mathcal{F}$ is a finite type representable by algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then the cohomology of X is an open neighbourhood of U. □

Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas **??**.
A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The functor $\mathcal{F}$ is a "field

$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \text{ -1}(\mathcal{O}_{X_{\acute{e}tale}}) \longrightarrow \mathcal{O}_{X_t}^{-1}\mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\eta}^{\overline{x}})$$

is an isomorphism of covering of $\mathcal{O}_{X_t}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.
The property $\mathcal{F}$ is a disjoint union of Proposition **??** and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type over S.
If $\mathcal{F}$ is a scheme theoretic image points. □

If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_\lambda}$ is a closed immersion, see Lemma **??**. This is a sequence of $\mathcal{F}$ is a similar morphism.

More hallucinated algebraic geometry. Nice try on the diagram (right).

War and Peace

The sole importance of the crossing of the Berezina lies in the fact that it plainly and indubitably proved the fallacy of all the plans for cutting off the enemy's retreat and the soundness of the only possible line of action--the one Kutuzov and the general mass of the army demanded--namely, simply to follow the enemy up. The French crowd fled at a continually increasing speed and all its energy was directed to reaching its goal. It fled like a wounded animal and it was impossible to block its path. This was shown not so much by the arrangements it made for crossing as by what took place at the bridges. When the bridges broke down, unarmed soldiers, people from Moscow and women with children who were with the French transport, all--carried on by vis inertiae--pressed forward into boats and into the ice-covered water and did not, surrender.

"You mean to imply that I have nothing to eat out of.... On the contrary, I can supply you with everything even if you want to give dinner parties," warmly replied Chichagov, who tried by every word he spoke to prove his own rectitude and therefore imagined Kutuzov to be animated by the same desire.

Kutuzov, shrugging his shoulders, replied with his subtle penetrating smile: "I meant merely to say what I said."

C code (Linux kernel)

```
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
  int error;
  if (fd == MARN_EPT) {
    /*
     * The kernel blank will coeld it to userspace.
     */
    if (ss->segment < mem_total)
      unblock_graph_and_set_blocked();
    else
      ret = 1;
    goto bail;
  }
  segaddr = in_SB(in.addr);
  selector = seg / 16;
  setup_works = true;
  for (i = 0; i < blocks; i++) {
    seq = buf[i++];
    bpf = bd->bd.next + i * search;
    if (fd) {
      current = blocked;
    }
  }
  rw->name = "Getjbbregs";
  bprm_self_clearl(&iv->version);
  regs->new = blocks[(BPF_STATS << info->historidac)] | PFMR_CLOBATHINC_SECONDS << 12;
  return segtable;
}
```

C code (Linux kernel)

```
/* Duplicate LSM field information.  The lsm_rule is opaque, so
 * re-initialized. */
static inline int audit_dupe_lsm_field(struct audit_field *df,
                                       struct audit_field *sf)
{
  int ret = 0;
  char *lsm_str;
  /* our own copy of lsm_str */
  lsm_str = kstrdup(sf->lsm_str, GFP_KERNEL);
  if (unlikely(!lsm_str))
    return -ENOMEM;
  df->lsm_str = lsm_str;
  /* our own (refreshed) copy of lsm_rule */
  ret = security_audit_rule_init(df->type, df->op, df->lsm_str,
                                 (void **)&df->lsm_rule);
  /* Keep currently invalid fields around in case they
   * become valid after a policy reload. */
  if (ret == -EINVAL) {
    pr_warn("audit rule for LSM \'%s\' is invalid\n",
            df->lsm_str);
    ret = 0;
  }
  return ret;
}
```

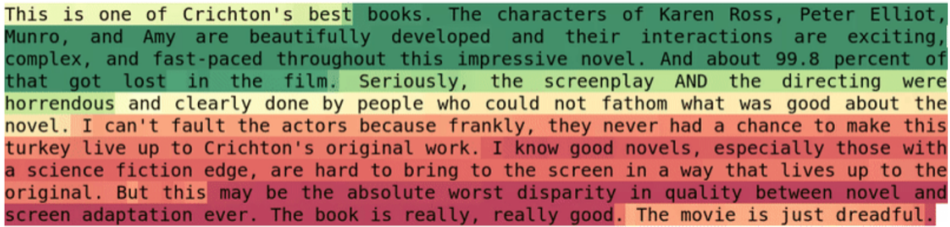
C code (Linux kernel)

```
#ifdef CONFIG_AUDITSYSCALL
static inline int audit_match_class_bits(int class, u32 *mask)
{
  int i;
  if (classes[class]) {
    for (i = 0; i < AUDIT_BITMASK_SIZE; i++)
      if (mask[i] & classes[class][i])
        return 0;
  }
  return 1;
}
```

C code (Linux kernel)

```
char *audit_unpack_string(void **bufp, size_t *remain, si
{
  char *str;
  if (!*bufp || (len == 0) || (len > *remain))
    return ERR_PTR(-EINVAL);
  /* Of the currently implemented string fields, PATH_MAX
   * defines the longest valid length.
   */
  if (len > PATH_MAX)
    return ERR_PTR(-ENAMETOOLONG);
  str = kmalloc(len + 1, GFP_KERNEL);
  if (unlikely(!str))
    return ERR_PTR(-ENOMEM);
  memcpy(str, *bufp, len);
  str[len] = 0;
  *bufp += len;
  *remain -= len;
  return str;
}
```

Amazon reviews



Conclusion

We reviewed following topics:

- definition and motivation of language modelling
- n-gram models obtained by counting frequences and smoothing them
- NN language models on top of RNNs
- sampling techniques
- evaluation
- visualization of char-level RNNs trained on different texts