Kodery. Reprezentacje wektorowe tokenów.

Paweł Rychlikowski

Instytut Informatyki UWr

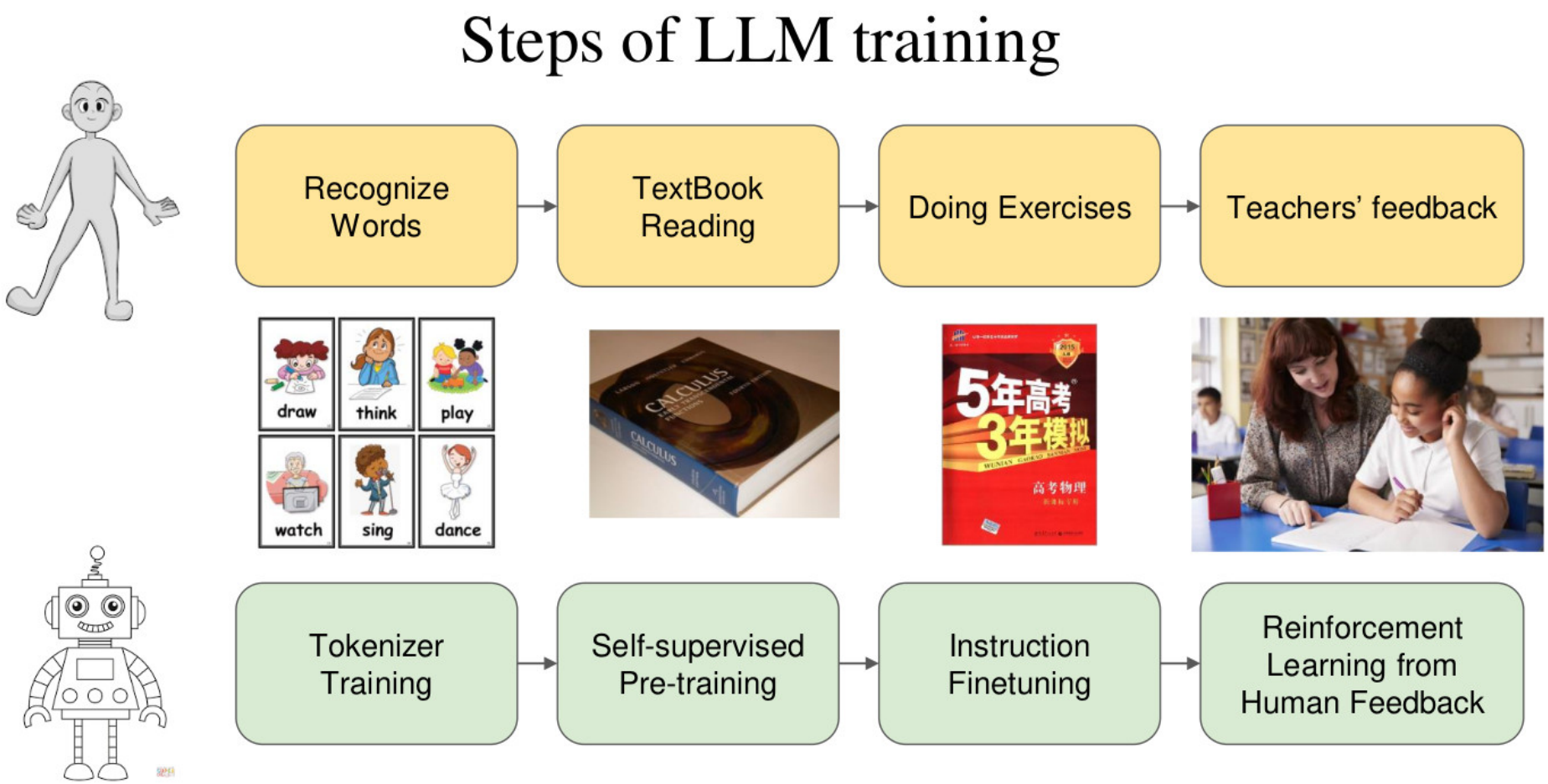
18 listopada 2024

Trening modelu. Przypomnienie

Trening można przeprowadzać w wielu etapach, na różnych korpusach, przykładowo:

- **Etap 1**: Trening wstępny (pretraining), na dużych korpusach tekstowych
- **Etap 2**: Trening na danych dziedzinowych (mniejsze korpusy, bardziej „na temat")
- **Etap 3**: Trening zadania docelowego (np klasyfikacji tekstów), ew. uczenie ze wzmocnieniem (jak mamy funkcję oceniającą generator)

Etapy uczenia (obrazek)



Etap 2: dotrenowywanie modelu

Uwaga

Wymaga mniejszego budżetu na trening, możemy dotrenowywać wiele modeli bazujących na tym samym modelu wstępnie wytrenowanym.

Inne opcje:

- (teksty konkretnego autora, albo z jakiejś epoki literackiej)
- Teksty dialogów (tworzenie chatbota)
- Teksty medyczne, ustawy i teksty prawnicze, (teksty dziedzinowe)

Etap 2: dotrenowywanie modelu (cd)

- Dialogi napisane przez ludzi przedstawiające wzorcowe działanie naszego bota (jeden z etapów treningu ChatGPT)
- Dane wygenerowane przez programy, na przykład dowody, rozumowania
  - Praca mgr: napisać generator zadań z rozwiązaniami, użyć do treningu
- Pary: specyfikacja-program
  - Generowanie kodu jest na tyle ważne, że poświęcimy mu osobny wykład

Etap 3. Uczenie ze wzmocnieniem

Funkcja oceniająca wygenerowane zdanie może uwzględniać:

- Długość zdań, różne właściwości gramatyczne tekstu
- Miary prostoty i czytelności tekstu (Pracownia Prostej Polszczyzny)
- Brak powtórzeń
- Naturalność tekstu (np. perplexity)
- Zgodność z faktami (TODO: jak ją mierzyć)
- Miary takie jak: stopień zgodności ze specyfikacją sonetu (sylaby, rymy, zwrotki)

Takie zabawy wymagają dużej uwagi: sztuczna funkcja oceny może mieć nieoczekiwane ekstrema.

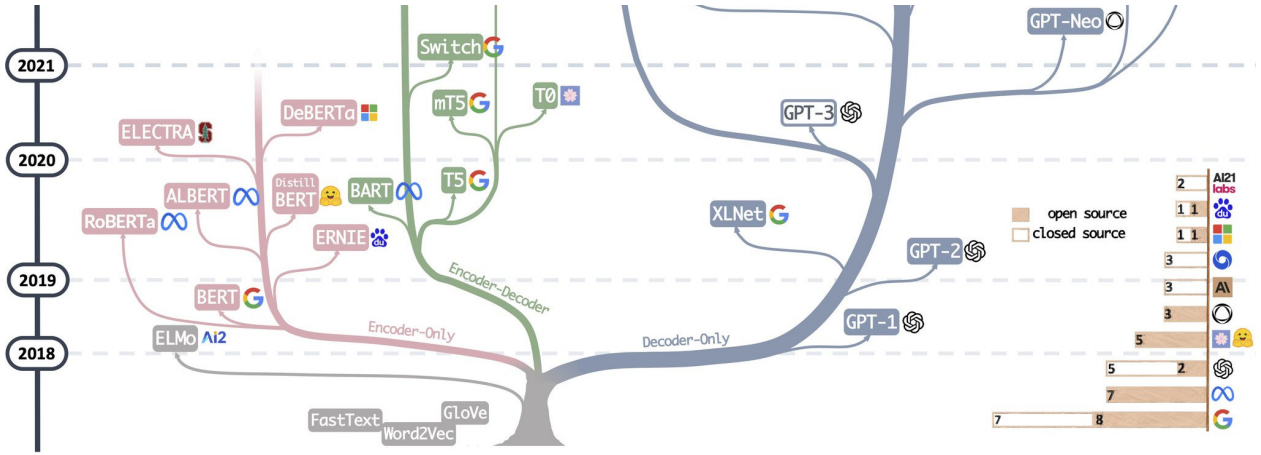
Przerwa na reklamę

Limeryk Wisławy Szymborskiej

Na przedmieściach żył Singapuru
pewien słynny z przemówień guru.
Lecz raz wyznał mi z płaczem:
„ Mam kłopoty z wołaczem
i do szczura wciąż mówię: szczuru!”

- Może projekt, albo praca dyplomowa o układaniu limeryków (lub innych utworów poezji ze ścisłymi wymogami formalnymi)? (nad podobnym tematem zastanawia się p. Michał)
- Haiku też się nada, ale raczej na miniprojekt...
- Z kolei piosenki disco-polo, połączone z kompozycją i syntezą śpiewu to raczej ambitna praca mgr

Różne rodzaje modeli językowych



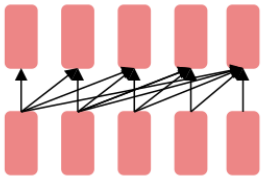
Trzy główne gałęzie (od lewej) to:

- Tylko **koder** (BERT)
- **koder-dekoder** (pierwszy transformer, T5, systemy tłumaczące)
- Tylko **dekoder** (GPT-x)

No i minigałązka zawierająca word2vec, fasttext, GloVe i ELMo

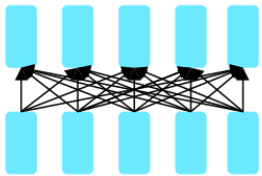
Pretraining for three types of architectures

The neural architecture influences the type of pretraining, and natural use cases.



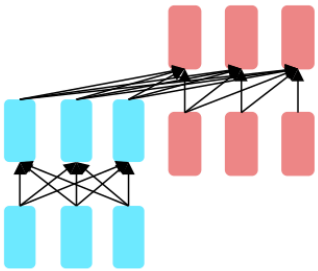
Decoders

- Language models! What we've seen so far.
- Nice to generate from; can't condition on future words
- **Examples:** GPT-2, GPT-3, LaMDA



Encoders

- Gets bidirectional context – can condition on future!
- Wait, how do we pretrain them?
- **Examples:** BERT and its many variants, e.g. RoBERTa



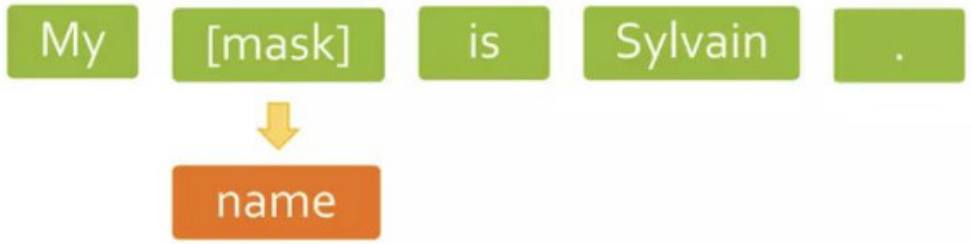
Encoder-Decoders

- Good parts of decoders and encoders?
- What's the best way to pretrain them?
- **Examples:** Transformer, T5, Meena

27

Autoregresywne i maskowane modele językowe

Guess the next word in the sentence (GPT)



Guess some masked words in the sentence (BERT)

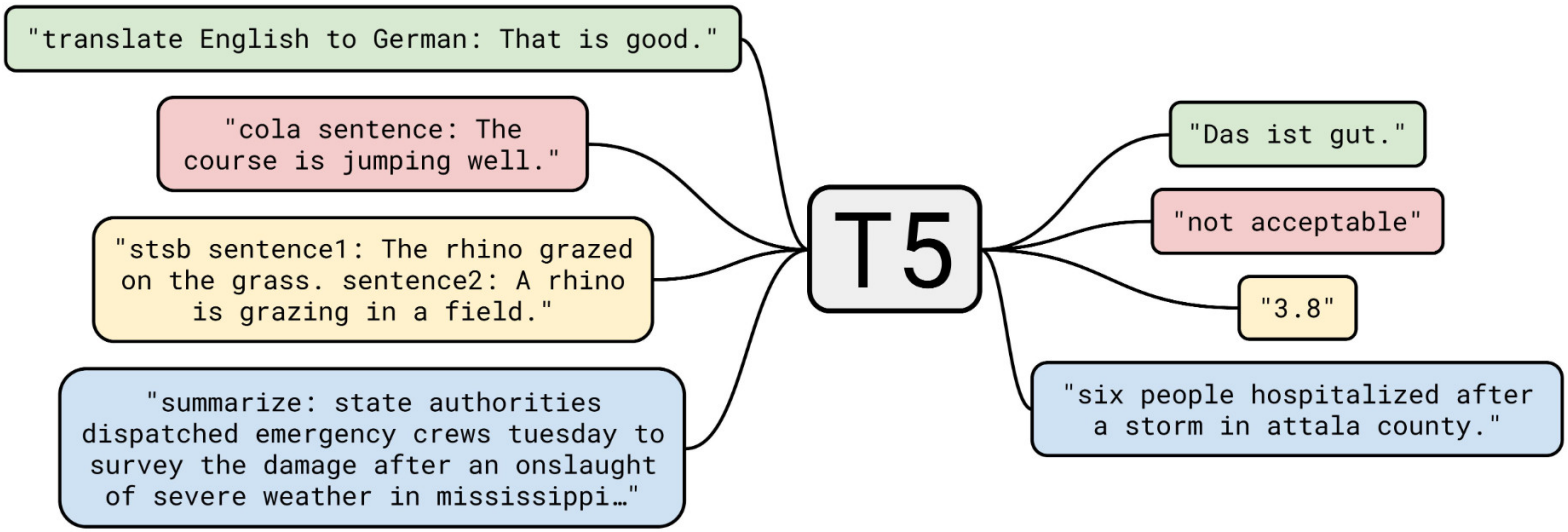
To drugie zadanie jest podstawowym zadaniem w treningu koderów!

Dwa podstawowe zadania w treningu transformera BERT

1. Maskowany model językowy:
   - 15% tokenów jest **maskowanych**
   - Dzielimy je na 3 części:
     - ★ 80% – zamieniamy na specjalny token [MASK]
     - ★ 10% – pozostawiamy bez zmian
     - ★ 10% – zamieniamy na losowy token
   - Zadaniem jest przewidzenie oryginalnego tokenu na tych wybranych pozycjach
2. Czy dwa zdania są sąsiadami w rzeczywistym tekście
   - Trochę za łatwe (dlaczego?)
   - Lepsza wersja: te zdania **są** sąsiednie, pytanie czy w dobrej kolejności (ALBERT)

Trening wstępny i zasadniczy modeli typu koder-dekoder

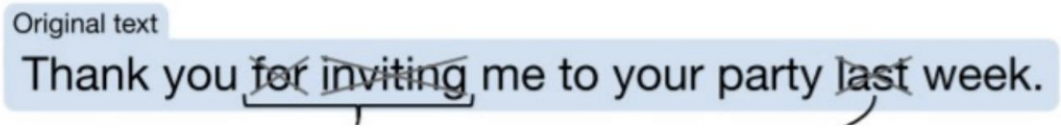
T5 – model ogólnego zastosowania, typu koder-dekoder (czyli znajduje reprezentację dla wejścia, generuje wyjście)



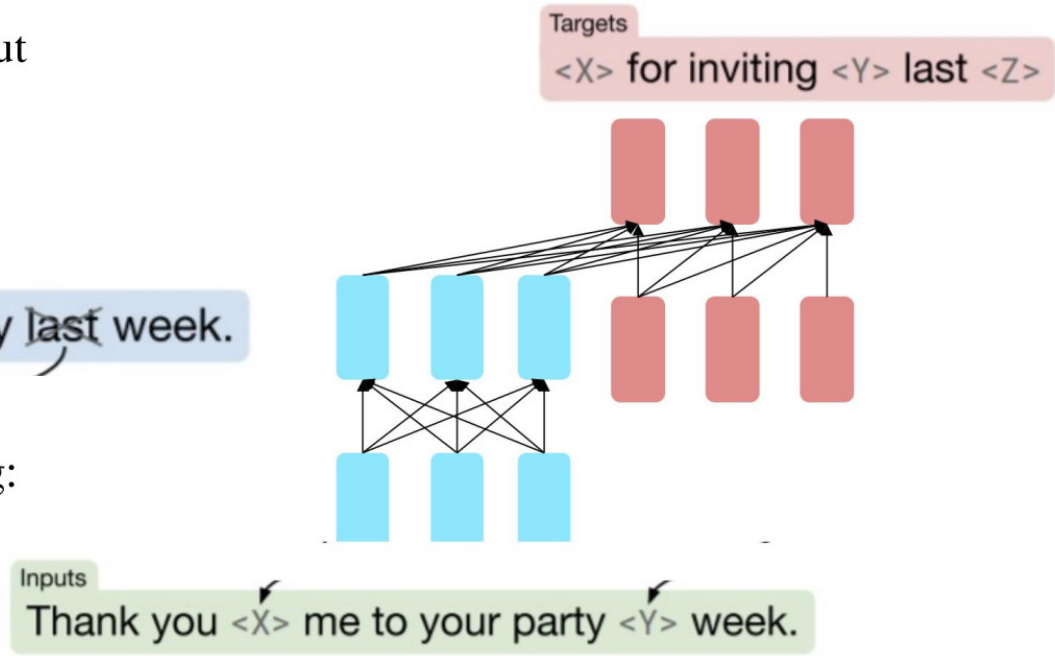
Pretraining Encoder-Decoders: Span Corruption

What [Raffel et al., 2018] found to work best was span corruption. Their model: T5.

Replace different-length spans from the input with unique placeholders; decode out the spans that were removed!



This is implemented in text preprocessing: it's still an objective that looks like **language modeling** at the decoder side.



Plan na dalsze wykłady

1. Osadzenia (zanurzenia, ...) słów w przestrzeni R^N (na przykładzie word2vec, ale nie tylko)
2. Ogólny schemat działania sieci transformer i osadzenia kontekstowe
3. NLP na transformerach typu BERT:
   - Klasyfikacja tekstów
   - Klasyfikacja tokenów (Named Entity Recognition, POS-tagging)
   - Czytanie ze zrozumieniem
   - Streszczanie
   - Tłumaczenie maszynowe
4. W końcu: zaimplementujemy transformery, wytrenujemy małe transformerki dla jakiegoś zadania.

Plan na pracownie

- **Pracownia 1**: Standardowa generacja tekstów, ocena prawdopodobieństwa tekstu
- **Pracownia 2**: Własna generacja tekstu – wymuszanie właściwości tekstu
- **Pracownia 3**: Zadania NLP z użyciem sieci BERT (Bidirectional Encoder Representations from Transformers), badanie reprezentacji słów i zdań, prosty RAG
- **Pracownia 4**: Gry, RL, QA, Transformer++, Toolformer (coś wybierzemy)

RAG == Retrieval Augmented Generation

Zanurzenia rzadkie i gęste

Definicja

Zanurzenie (embedding, osadzenie) odwzorowanie przestrzeni dyskretnej (zbioru skończonego, słów albo tokenów) w ciągłą przestrzeń R^n

Zanurzenia mogą być:

- **Rzadkie** – nieliczne niezerowe wartości, $n \approx 10^6$
- **Gęste** – 0 nie jest jakąś specjalną wartością, $n \approx 10^3$

Representing words as discrete symbols

In traditional NLP, we regard words as discrete symbols:

hotel, conference, motel – a localist representation

Means one 1, the rest 0s

Such symbols for words can be represented by one-hot vectors:

motel = [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]

hotel = [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]

Vector dimension = number of words in vocabulary (e.g., 500,000+)

16

Problem with words as discrete symbols

Example: in web search, if a user searches for "Seattle motel", we would like to match documents containing "Seattle hotel"

But:

motel = [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]
hotel = [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]

These two vectors are orthogonal

There is no natural notion of **similarity** for one-hot vectors!

Solution:

- Could try to rely on WordNet's list of synonyms to get similarity?
  - But it is well-known to fail badly: incompleteness, etc.
- **Instead: learn to encode similarity in the vectors themselves**

17

Znaczenie słowa

Pytania

1. Czym jest znaczenie słowa?
2. Kiedy znamy znaczenie słowa?
3. Jak wiele tekstów nam potrzeba, żeby to znaczenie odgadnąć.

- Rościan wykorzystywany jest przede wszystkim w kuchni, gdzie oprócz spożywania na surowo jest również gotowany i smażony.
- Stare przysłowie głosi: „Szczi i kasza to pożywienie nasze”
- Dna moczanowa spowodowana jest odkładaniem w tkankach kryształów moczanu sodu.

Znaczenie słowa

Co wiemy o znaczeniu słowa:

- Wynika z kontekstu.
- Kontekstem są słowa (pewne?, wszystkie?) z którymi współwystępuje nasze słowo.

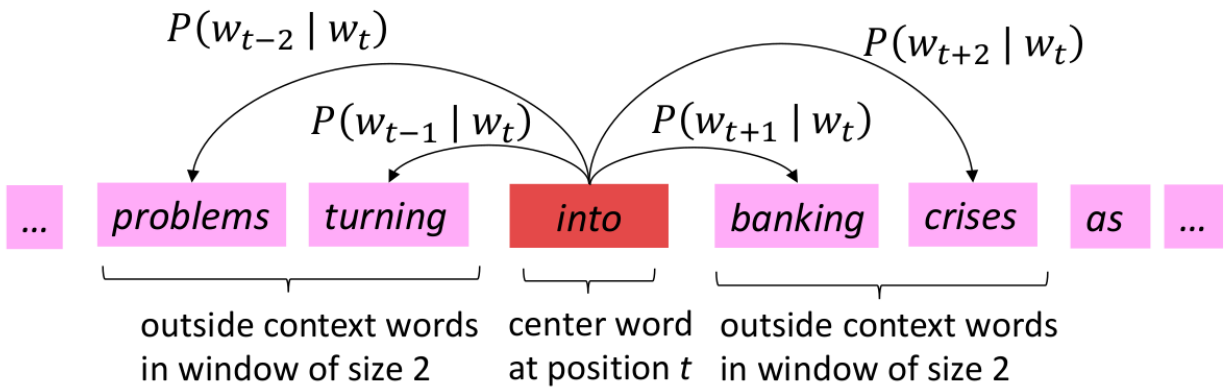
Uwaga

Część znaczenia słowa można wywnioskować z jego budowy:

- dendro**logia** – znaczący sufiks
- drzewoznawstwo – znaczące dwie części słowa
- szczi – pojęcie *egzotyczne*
- **anty**konsumpcjonizm – znaczący prefiks
- ...

Word2Vec Overview

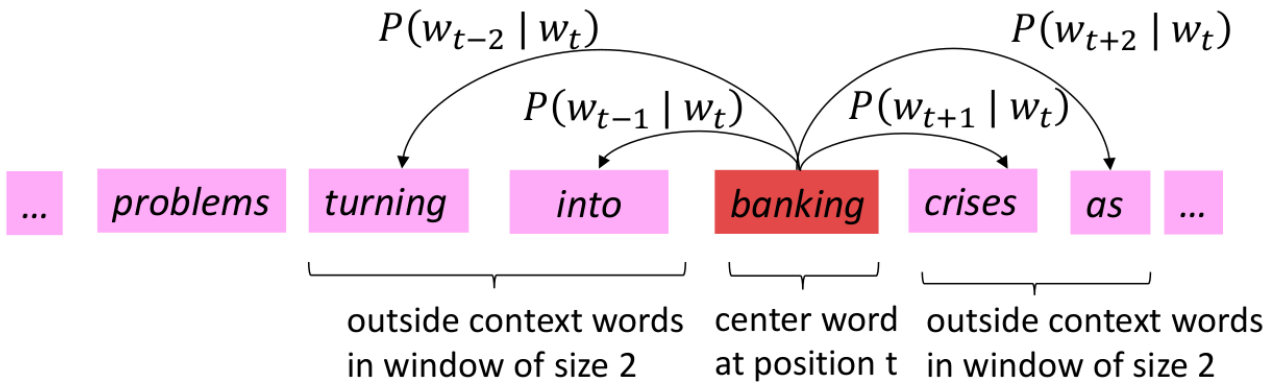
Example windows and process for computing $P(w_{t+j} \mid w_t)$

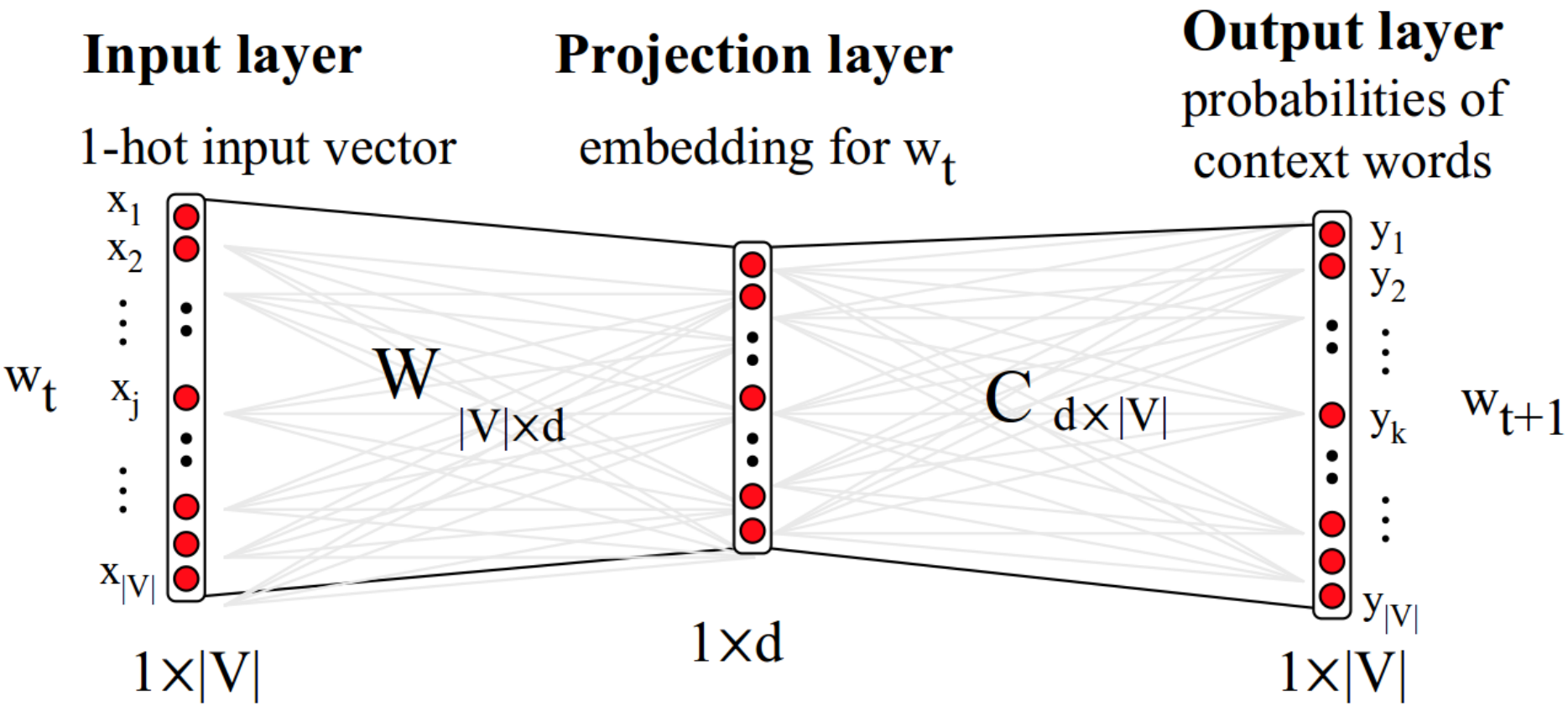


22

Word2Vec Overview

Example windows and process for computing $P(w_{t+j} \mid w_t)$




Input layer
1-hot input vector
Projection layer
embedding for w_t
Output layer
probabilities of context words
w_t
x_1
x_2
x_j
$x_{|V|}$
$W_{|V|\times d}$
$C_{d\times|V|}$
y_1
y_2
y_k
$y_{|V|}$
w_{t+1}
$1\times|V|$
$1\times d$
$1\times|V|$

Prawdopodobieństwo słowa

- Używamy tzw. Softmax layer
- $$P(w_k|w_j) = \frac{\exp(c_k \cdot v_j)}{\sum_{i \in V} \exp(c_i \cdot v_j)}$$
- Ponieważ chcemy żeby to było duże, powinniśmy dążyć do:
  a) Zwiększenia licznika (c_k zbliża się do v_j)
  b) Zmniejszenia mianownika (inne c_i oddalają się od v_j)

Jak działa word2vec?

- Zbliżamy słowo do kontekstu, w którym występuje.
- Oddalamy słowo od kontekstów, w których nie wystąpiło.

Jak działa word2vec?

- Zbliżamy słowo do kontekstu, w którym występuje.
- Oddalamy słowo od kontekstów, w których nie wystąpiło.**(niektórych)**