

Neuronske mreže

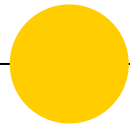
Character-level embedding



Agenda

- *Character-level embedding*
- *ELMo*
- Dodatno čitanje

Character-level embedding





Character-level embedding

- *Transfer learning* u *NLP*-u:
 - Da li je moguće napraviti univerzalan *Language model* koji bi se mogao koristiti u raznim *NLP* (*Natural Language Processing*) zadacima (kao što koristimo pretrenirane *VGGNet*, *ResNet*, ...)?
 - Više pokušaja istraživanja u ovom smeru.



Character-level embedding

- *RNN* - primer klasifikacije sekvence:
 - *One-hot encoding*
 - Svaka reč je imala svoj *word vector*, koji u sebi sadrži samo jednu jedinicu i $n-1$ nula, gde je n ukupan broj svih poznatih reči
 - *Word vector* smo dobijali tako što stavimo 1 na indeks date reči u rečniku svih poznati reči
 - Širina vektora = veličina rečnika
 - Jako loša osobina *One-hot encoding*-a



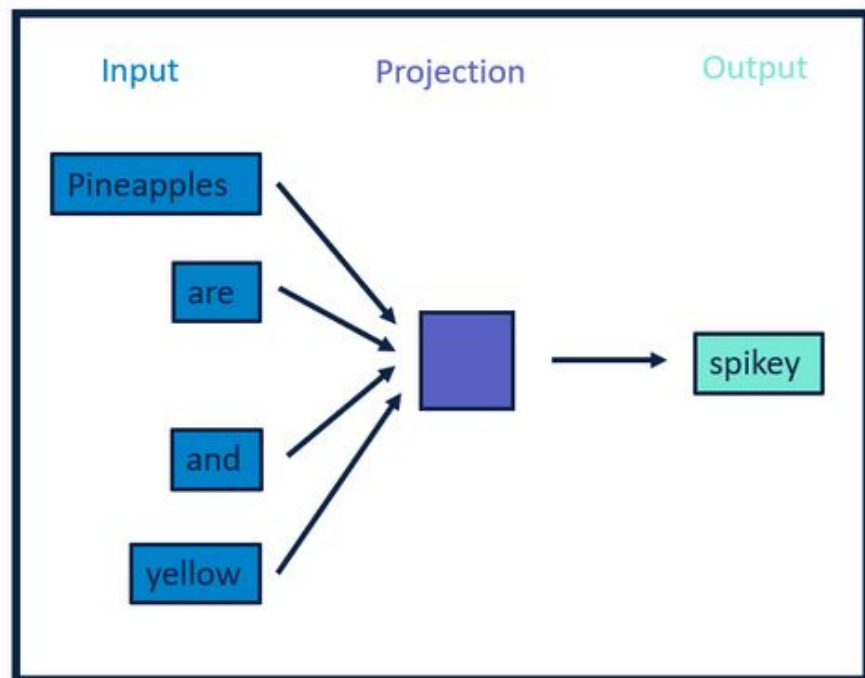
Character-level embedding

- *RNN* - primer klasifikacije sekvence:
 - Da li kod *one-hot encoding*-a imamo informaciju o tome koliko su određene reči “slične”?
 - Ako u rečniku imamo reči [“doberman”, “senka”, “vernost”, “dresiran”], *one-hot* nam neće dati informaciju o tome koje reči su potencijalno bliske.

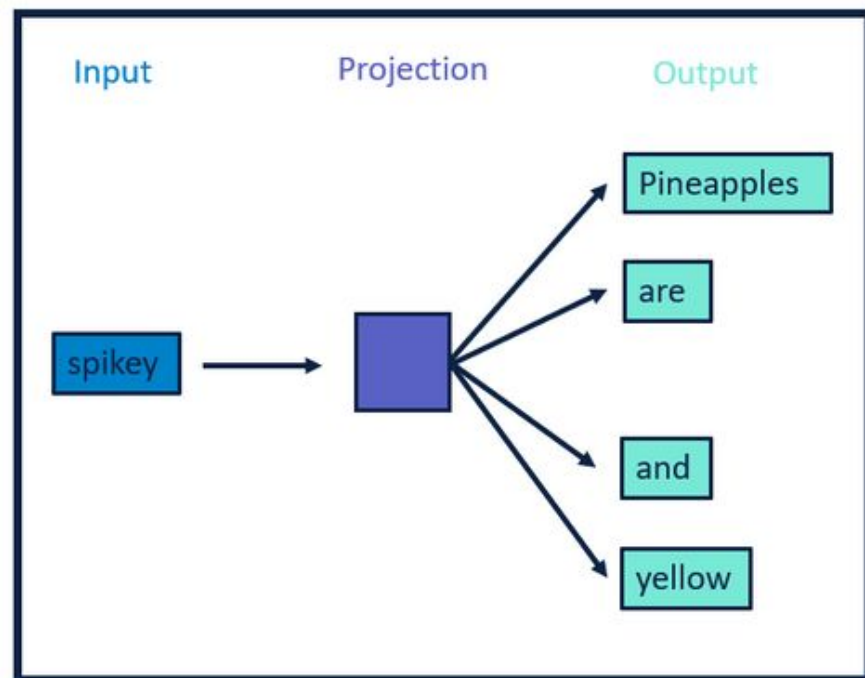


Character-level embedding

- *Word2Vec*:
 - Koncept prvi put predstavljen 2013. godine
 - Ideja je u tome da se ulazni *one-hot* (*sparse*) vektori projektuju na prostor mnogo manjih dimenzija (*dense* prostor), u kome će sami vektori sadržavati i neke osnovne informacije o semantici koju te reči nose
 - Dve implementacije:
 - *skip-gram*
 - *continuous bag of words (CBOW)*



CBOW



Skip-gram




Character-level embedding

- *Word2Vec*:
 - Problemi (1/2):
 - Korišćenjem *word embedding*-a podižemo tačnost modela, pošto smo uveli semantički bogatije vektore nižeg nivoa dimenzionalnosti, u odnosu na *one-hot* vektore.
 - Šta ako se pojavi reč koja ne postoji u rečniku?
 - Možda nam nije poznata
 - Možda je pogrešno napisana (*typo*)
 - Možda se samo nalazi u drugom obliku (recimo u množini)
 - ...



Character-level embedding

- Idealno bi bilo kada bi *embedding* gledao pojedinačna slova (karaktere) umesto kompletne reči
- Na taj način bi reči koje imaju manju morfološku promenu (jednina u množinu i slično) i dalje imale sličan *embedding*:
 - Ovo bi rešio i *Word2Vec*, ali ni on neće biti u mogućnosti ako je ta promena nepravilna i ne postoji u skupu podataka nad kojim je obučen
 - Tipičan primer su skraćenice, sleng (*whatev*, *m8*) ili *typo*.

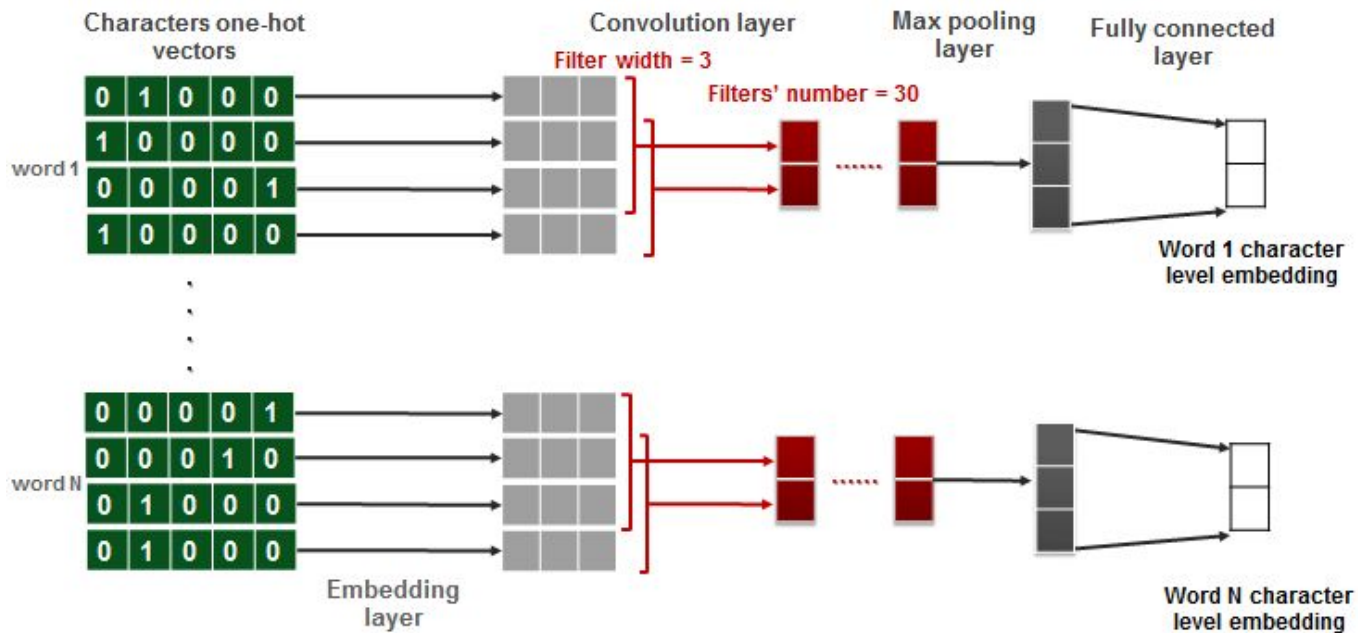


Character-level embedding

- Zato uvodimo *character-level embedding* i pokazaćemo dva načina na koje se on može realizovati:
 - *CNN*
 - *RNN*.

Character-level embedding

CNN



● Character-level embedding

● CNN

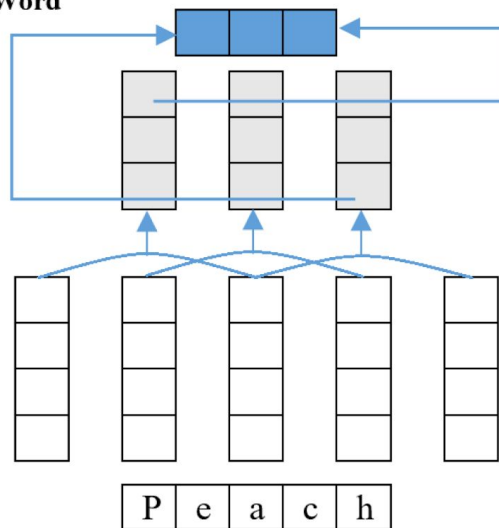
Character-level CNN-based word encoding

Representation of a Word

Global Max-Pooling
Convolution

Character
Embedding

Word



● **Character-level embedding**

- **CNN:**
 - Određuje koja slova su bitna za kreiranje reči
 - Primer u psihologiji je tipoglikemija

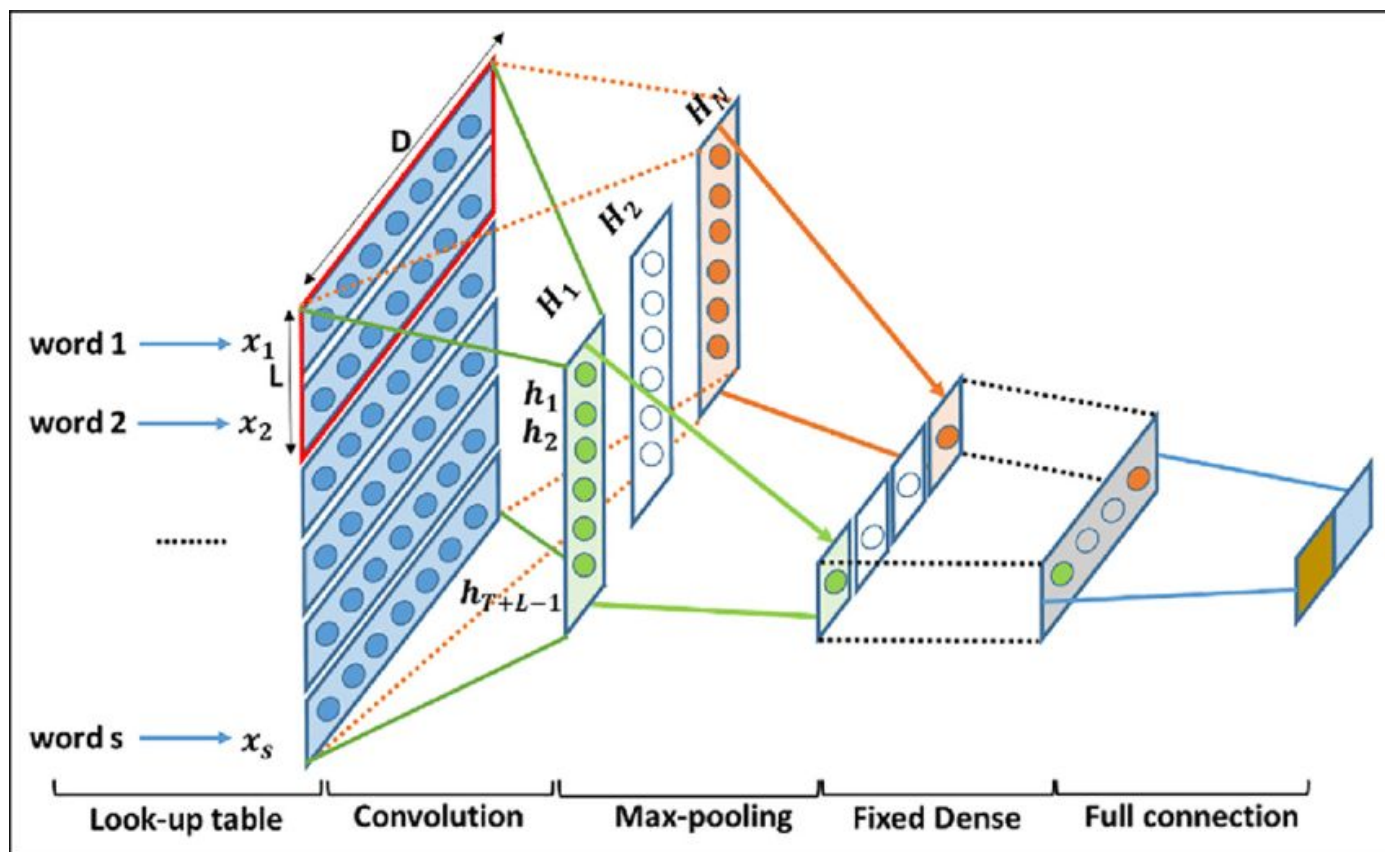
T1P06LIK3MIJ4
P0RUK4 M0Z3 B1T1 K0MB1N0V4N4 SA
BR0J3V1M4, AL1 AKO S3 M4L0 K0NC3NTR1S3TE
M0Z3TE SA L4K0C0M 0V0 D4 PR0C1T4T3

CNN takođe može da se fokusira na slova koja su nosioci reči i da ipak napravi dobar vektor, čak i kada imamo slovne greške, zahvaljujući osobinama konvolutivnih filtera *pooling* sloja.



Character-level embedding

- Da li se *CNN* mogu koristiti za klasifikaciju celog teksta?
 - Da.





Character-level embedding

- *RNN*

Character-level LSTM-based word encoding

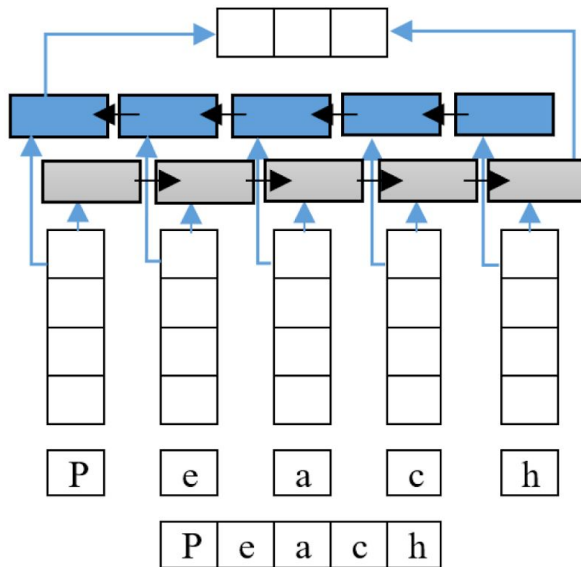
Representation of a Word

Backward
Long Short-Term Memory
(LSTM)

Forward
Long Short-Term Memory
(LSTM)

Character
Embedding

Word



Character-level embedding

- *Character-level embedding* nam rešava problem kada su reči pogrešno napisane ili kada su izvedene iz svog osnovnog oblika
- On će tehnički naći zajedničke osobine (kao što je koren reči) i dati dobar *embedding* i kada reč ima slovnu grešku ili je u neobičnom obliku
 - Ovo je odlično za tekstove sa dosta slenga, izvedenica ili slovnihih grešaka



Character-level embedding

- *Character-level embedding* nam ipak ne daje kontekst reči i šta ona znači
- Zato rečima moramo dati kontekst korišćenjem dodatnih modela i mehanizama.



Character-level embedding

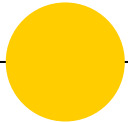
- *Word2Vec:*
 - Problemi (2/2):
 - Šta u slučaju kada imamo **homonime** kao reči u tekstu (isto se pišu, ali imaju različito značenje)?
 - Pas (životinja) i pas (pojas)
 - Kosa (na glavi) i kosa (linija)
 - Vredno je radio, pa je kupio radio.



Character-level embedding

- *Word2Vec*:
 - Problemi (2/2):
 - Šta ako reč ima slovnu grešku ili ne postoji u rečniku?
 - Do sada smo te reči obično menjali nekim predefinisanim vektorom (nula vektor ili slično) ili smo te reči izbacivali iz teksta jer *Word2Vec* ne zna šta će sa njima jer mu ništa ne znače.
 - *Word2Vec* se trenira jednom, tako se za svaku reč formira njen N-dimenzioni *dense* vektor
 - Ali kao što vidimo u primeru homonima, značenje reči se nekada ne može zaključiti samo na osnovu same reči, nego nam je i u *runtime*-u potreban kontekst.

ELMo

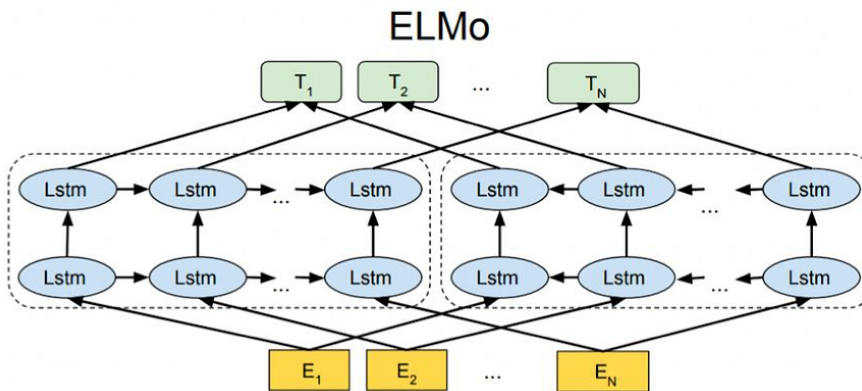


ELMo

- Prvi put objavljen u februaru 2018.



ELMO (1980)



ELMo (2018)



ELMo

- Problem prethodnih pristupa je *polysemy* (višeznačnost) reči, odnosno reči koje se pišu na isti način, ali se značenje razlikuje u odnosu na kontekst u kome se reči nalaze:
 - *I arrived at the **bank** after crossing the river.*
 - *I arrived at the **bank** after crossing the street.*

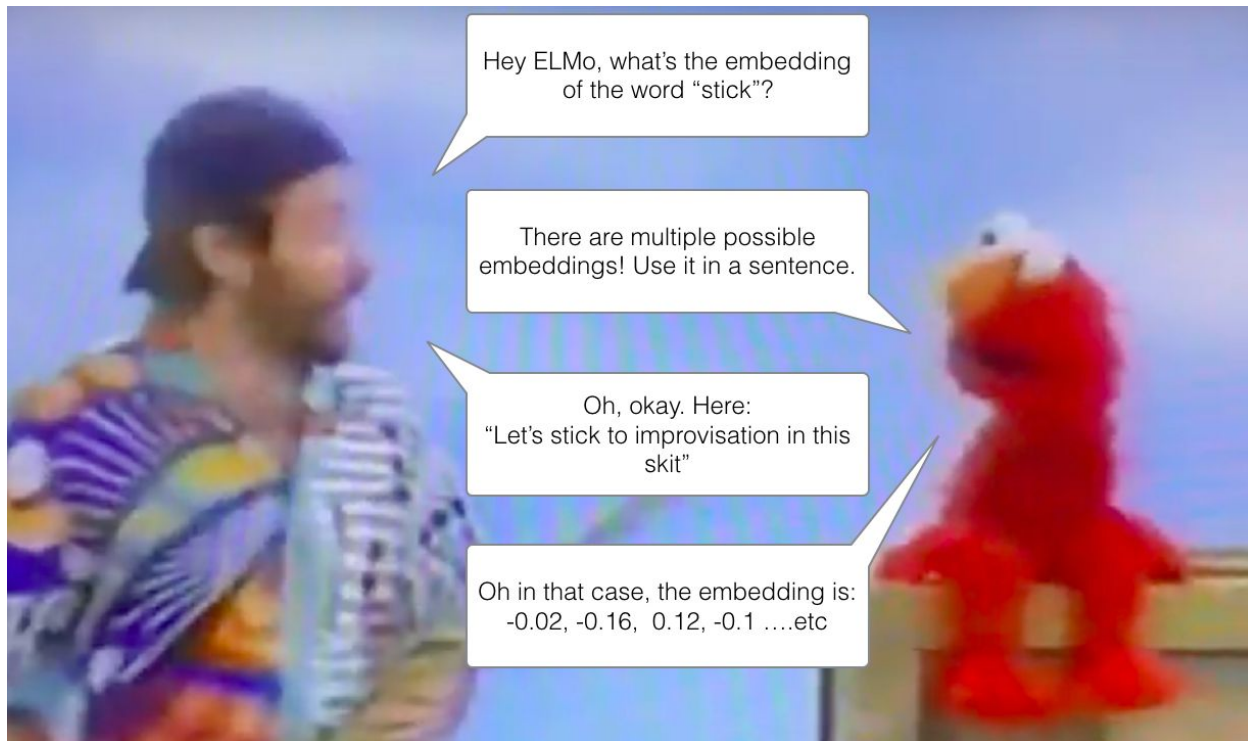


ELMo

- Osnovne osobine:
 - ELMo reprezentacija reči zavisi od celog konteksta u kome se reč koristi:
 - Ne može napraviti *embedding* jedne reči, nego mu treba cela rečenica za to
 - Rezultat su *word* vektori za svaku reč, ali su uslovljeni kontekstom (svaki put malo drugačiji *word* vektor)



ELMo



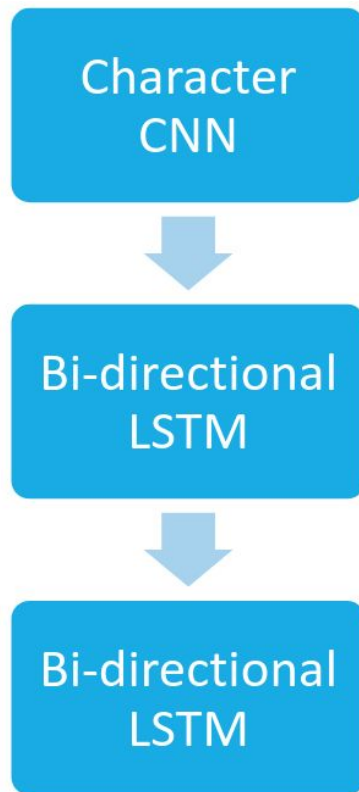


ELMo

- Osnovne osobine:
 - ELMo reprezentacija kombinuje sve slojeve duboke pretrenirane neuronske mreže (nije *shallow*)
 - *Character-based*:
 - Bazira se na morfologiji i može da funkcioniše čak i za reči koje nisu ranije viđene tokom treninga - nema ih u rečniku
 - *Character CNN*

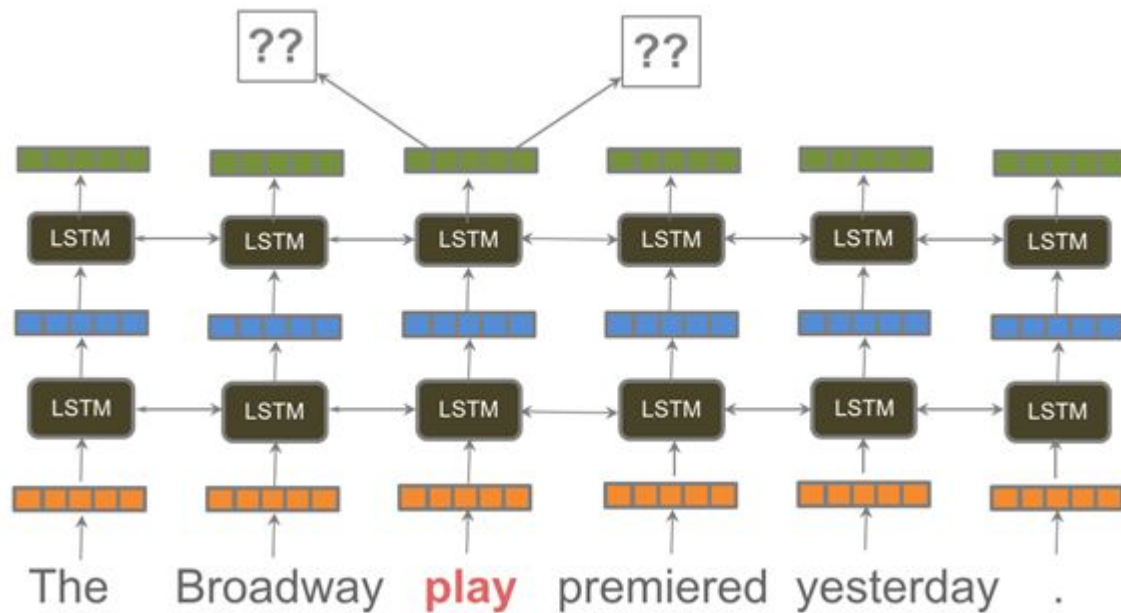
● **ELMo**

- Arhitektura:

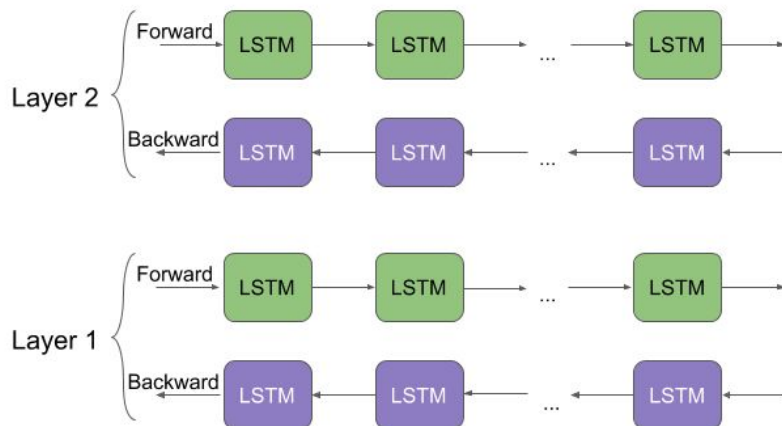


ELMo

- Arhitektura:



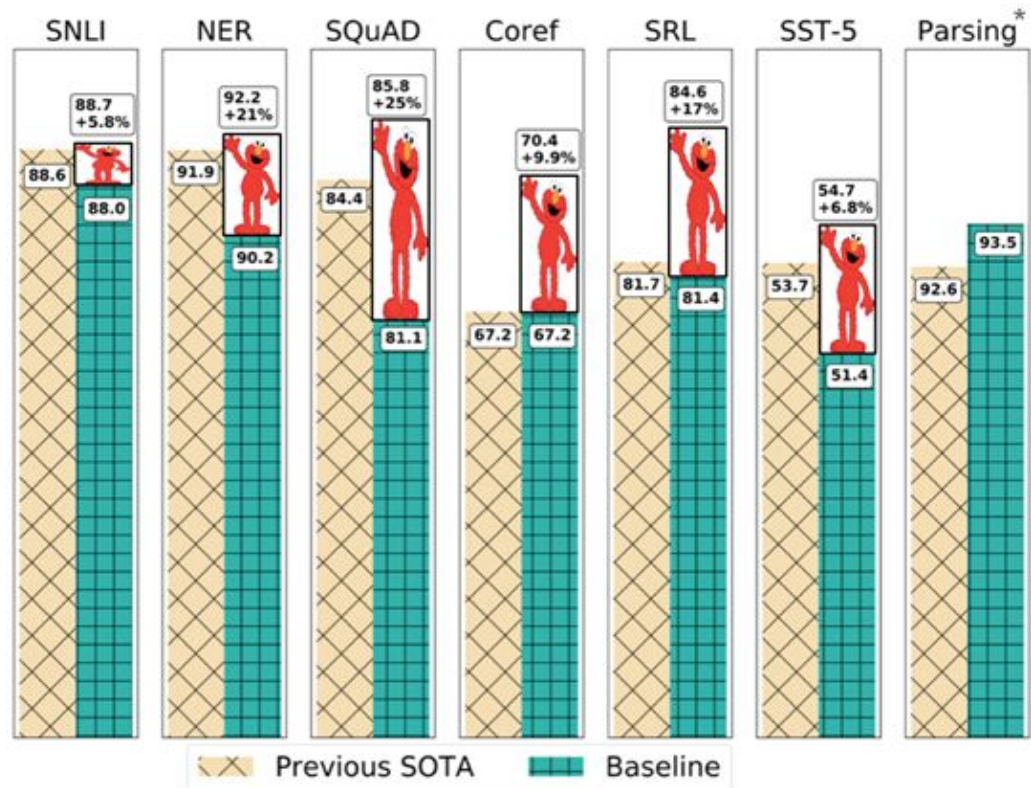
ELMo



- Koristi *character-level CNN* da predstavi reči i ovo je ulaz u prvi *BiLSTM* sloj
- *Forward pass* sadrži informacije o samoj reči i kontekstu pre te reči (levo od nje)
- *Backward pass* sadrži informacije o samoj reči i kontekstu posle te reči (desno od nje)
- Ovaj par informacija (*forward + backward*) formira *intermediate word* vektore koji onda ulaze u drugi sloj *BiLSTM*-a
- Konačna reprezentacija reči kroz *ELMo* se računa kao težinska suma ulaznih *word* vektora i izlaznih vektora oba sloja *LSTM*-a.

ELMo

- Rezultati:



*Kitaev and Klein, ACL 2018 (see also Joshi et al., ACL 2018)



- Kako napraviti dobar *embedding*?
 - Zavisi koji problem rešavamo
 - Nekim problemima treba manje kompleksan pristup (manje informacija o samoj reči), a nekim problemima treba više
 - Često se u praksi koristi kombinacija više *embedding*-a da bi se svaka reč opisala što bolje i *embedding*-u se dodaju različite meta-informacije



ELMo



- *Word2Vec/GloVe embedding, Character-level embedding, POS (Part-Of-Speech) embedding, NER (Named Entity Recognition) embedding, neki embedding definition od strane domenskog eksperta (recimo da li je reč česta, da li se često pojavljuje u domenu problema), sentiment embedding itd.*

ELMo

- *ELMo vs BERT:*

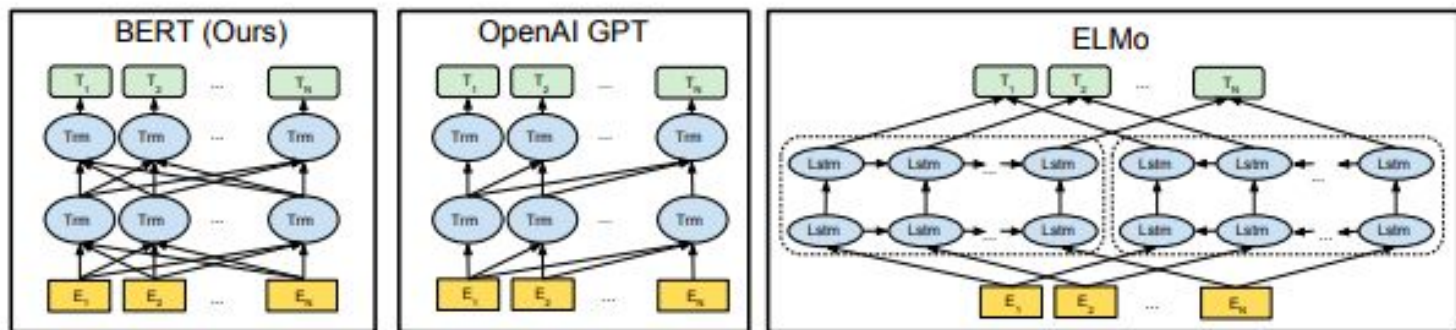
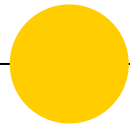


Figure 1: Differences in pre-training model architectures. BERT uses a bidirectional Transformer. OpenAI GPT uses a left-to-right Transformer. ELMo uses the concatenation of independently trained left-to-right and right-to-left LSTM to generate features for downstream tasks. Among three, only BERT representations are jointly conditioned on both left and right context in all layers.

Dodatno čitanje





Dodatno čitanje

- Deep contextualized word representations
- Create a Strong Text Classification with the Help from ELMo

Hvala na pažnji!

Pitanja?