SDS Tracker - Final Report

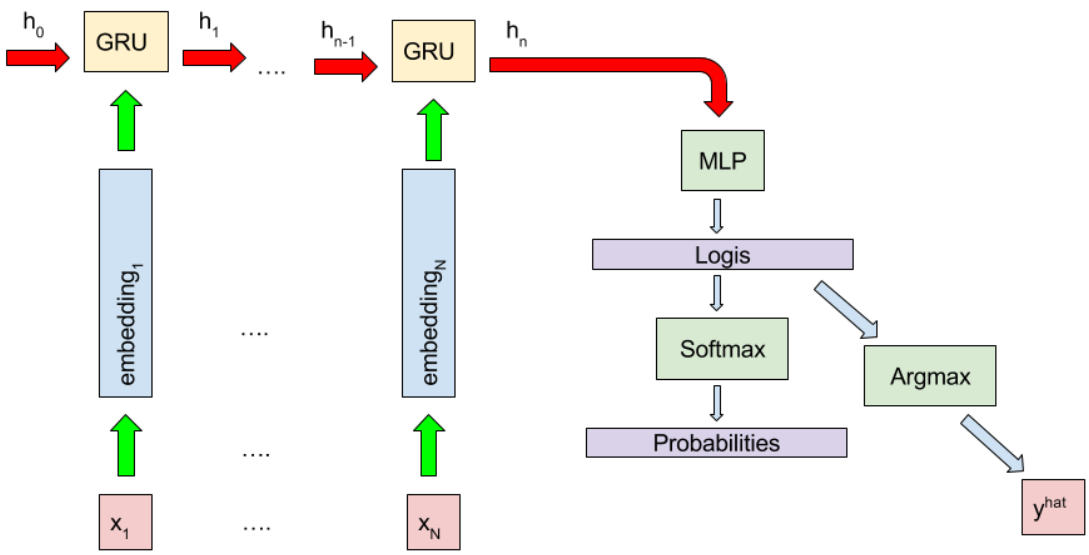
Petr Bělohlávek

Architektura

Dialog tracker se skládá ze základní jednotky X, která přečte jeden turn daného dialogu a předpoví label. X je dále používáno pro zpracování celého dialogu.

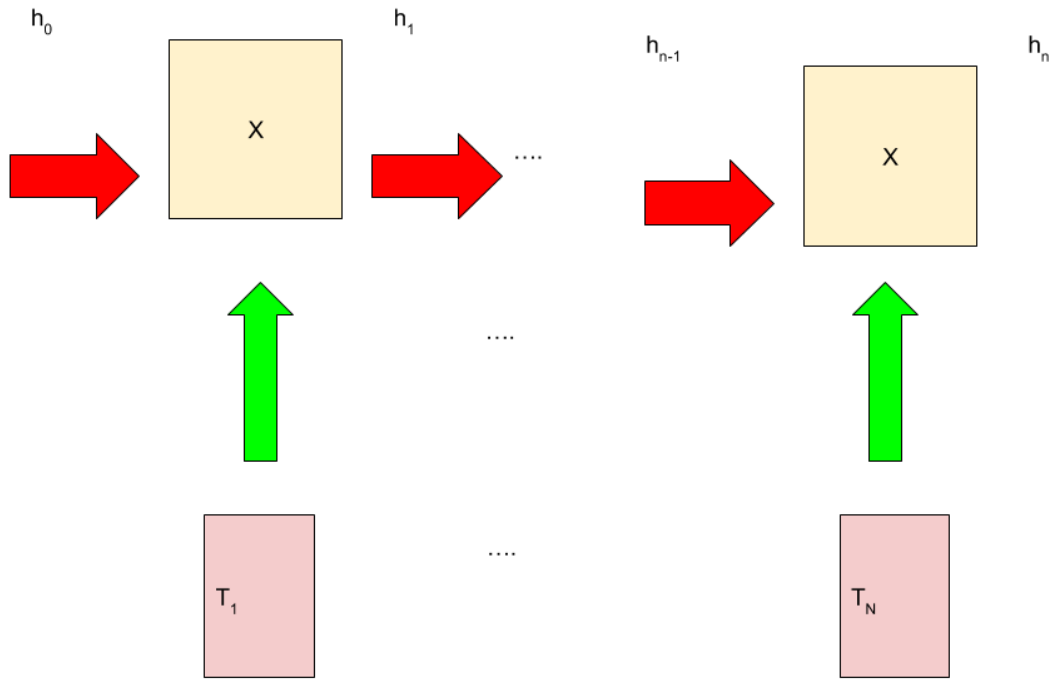
Turn-level (jednotka X)

Architektura X je následující. Vstup (jednotlivá slova turnu) se převedou na předem náhodně vygenerované embeddingy. Nad embeddingy následně scanuje jedna GRU jednotka a buduje si myšlenku (skrytý stav) nad textem.
Za posledním slovem je myšlenka předána do MLP (pro jednoduchost jsme zkusili i pouze projekční matici), který vrátí tzv. logits, tedy vektor čísel, který je stejně dlouhý jako počet všech labelů.
Nad logits dále spouštíme softmax, jehož výstupem je pravděpodbnostní distribuce nad jednotlivými labely. Label s maximální hodnotou (stačí na logits) je výstupem sítě.
Jako cost funkci uvažujeme categorial cross entropy, která se na klasifikaci hodí nejvíce. Ta kromě logits bere cílenou pravděpodobnostní distribuci, což je v našem případě one-hot encoding čísla skutečného labelu.



Dialog-level

Samotný dialog je zpracováván přes jednotlivé turny. Počáteční stav je trénovatelný parametr. Ten se předá do jednotky X společně s prvním turnem a dále se iteruje přes všechny ostatní turny.



Maskování

Celý trénink probíhá po batchích, tudíš je nutné vstupy maskovat a to na dvou úrovních. Nejdříve na dialog-level, kde dialogy v jednom batchi mohou mít různý počet turnů. Dále pak v batchi i-tých turnů, které mohou mít odlišný počet slov.

Baseline

Data obsahují celkem téměř 900 labelů. První z nich však zaobírá víc jak 22%. Z čistě klasifikačního hlediska toto považuji za triviální baseline, přestože jsou bezpochyby jiné přísnější.

Experimenty

Defaultní nastavení (embedding=300, hiddenstate=200) má dostatečně mnoho parametrů a train cost vytrvale klesá. Generalizace je ale velmi slabá a přes 23% accuracy jsem se nedostal. Model se pkraticky učí na neznámých datech předpovídat nulu.

Menší počet parametrů zapříčinil drobný nárůst accuracy (cca 25%), což naznačuje, že model se v předchozím případě už od začátku overfitoval. Navíc defaultní learning rate je příliš vysoká a loss zbytečně osciluje. Její zmenšení problém vyřešilo.

Menší batch z nějakého důvodu zlepšuje výrazně accuracy (kolem 32%)

Future Work

V krátkých bodech uvádím, co považuji za důležité na vyzkoušení a pro napsání článku:

- pořádně zkontrolovat, jak měříme accuracy
- zkusit jiné učící pravidla, ideálně taková, která si samy adaptují learning rate
- zkontrolovat, že celá architektura je opravdu dobře zapojená a nestává se např. ignorace vstupu
- zkontrolovat maskování
- při permutaci vstupu by nebylo špatné načíst vždy několik batchů dopředu a setřídit si je podle délky, což náhodnost moc nezmění a může to pomoci výkonu
- odhalit, proč velikost batche tolik ovlivňuje accuracy (tohle je opravdu podezřelé a stojí za vysvětlení, možná je tam někde velká chyba)

Závěr

Naimplementovali jsme netriviální model pro dialog tracking pomocí hlubokých rekurentních sítí v TensorFlow. Model se na trénovacích datech učí dobře, na validačních příliš ne, nicméně v konkrétních případech překonává námi nastavenou baseline signifikantně.