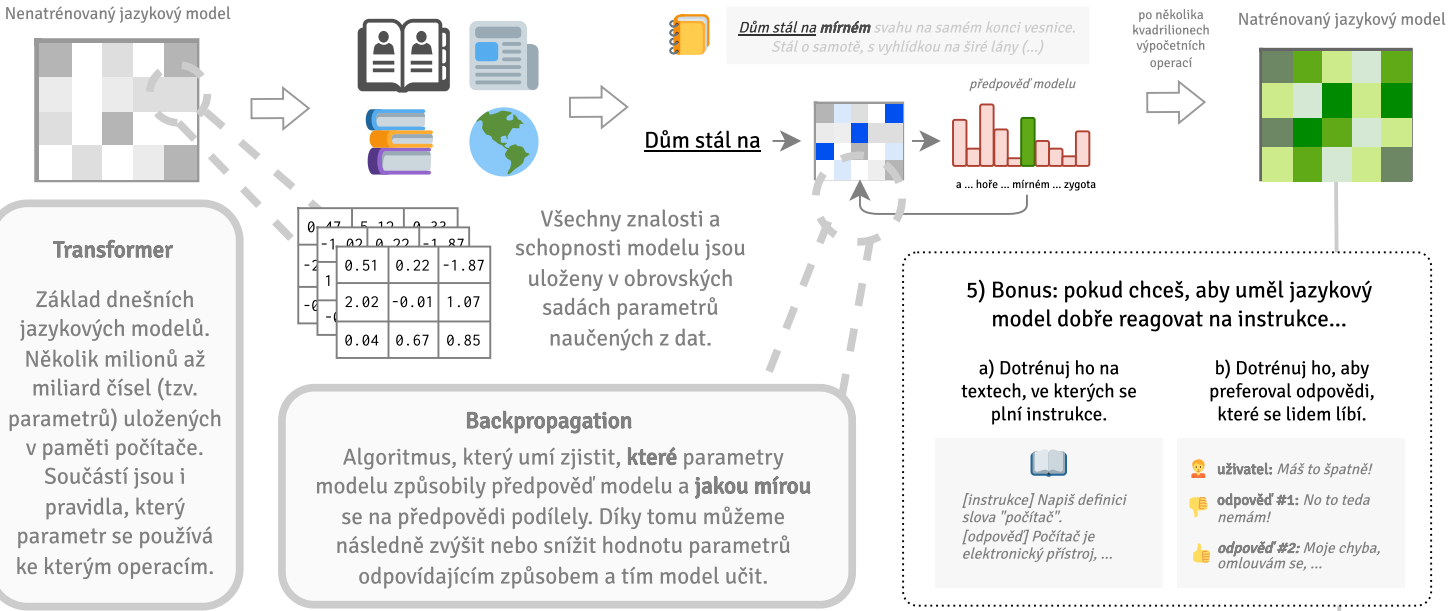
Recept: Jak vytvořit velký jazykový model

1) Vezmi Transformer a nastav mu náhodně parametry. (Pokud chceš velký jazykový model, vezmi opravdu velký Transformer.)

2) Nasbírej spousty textů: z internetu, z dokumentů, z knih. Čím více textů a čím kvalitnějších, tím lépe.

3) Pro každý kus textu se zeptej modelu, které slovo by mělo následovat. Když se netrefí do správného slova, lehce uprav hodnoty příslušných parametrů tak, aby slovo pro tento kus textu předpovídal s vyšší pravděpodobností.

4) Hotovo! Náš model umí obstojně předvídat další slovo ve větě (alespoň v podobných textech).



Transformer

Základ dnešních jazykových modelů. Několik milionů až miliard čísel (tzv. parametrů) uložených v paměti počítače. Součástí jsou i pravidla, který parametr se používá ke kterým operacím.

Všechny znalosti a schopnosti modelu jsou uloženy v obrovských sadách parametrů naučených z dat.

Backpropagation

Algoritmus, který umí zjistit, **které** parametry modelu způsobily předpověď modelu a **jakou mírou** se na předpovědi podílely. Díky tomu můžeme následně zvýšit nebo snížit hodnotu parametrů odpovídajícím způsobem a tím model učit.

5) Bonus: pokud chceš, aby uměl jazykový model dobře reagovat na instrukce...

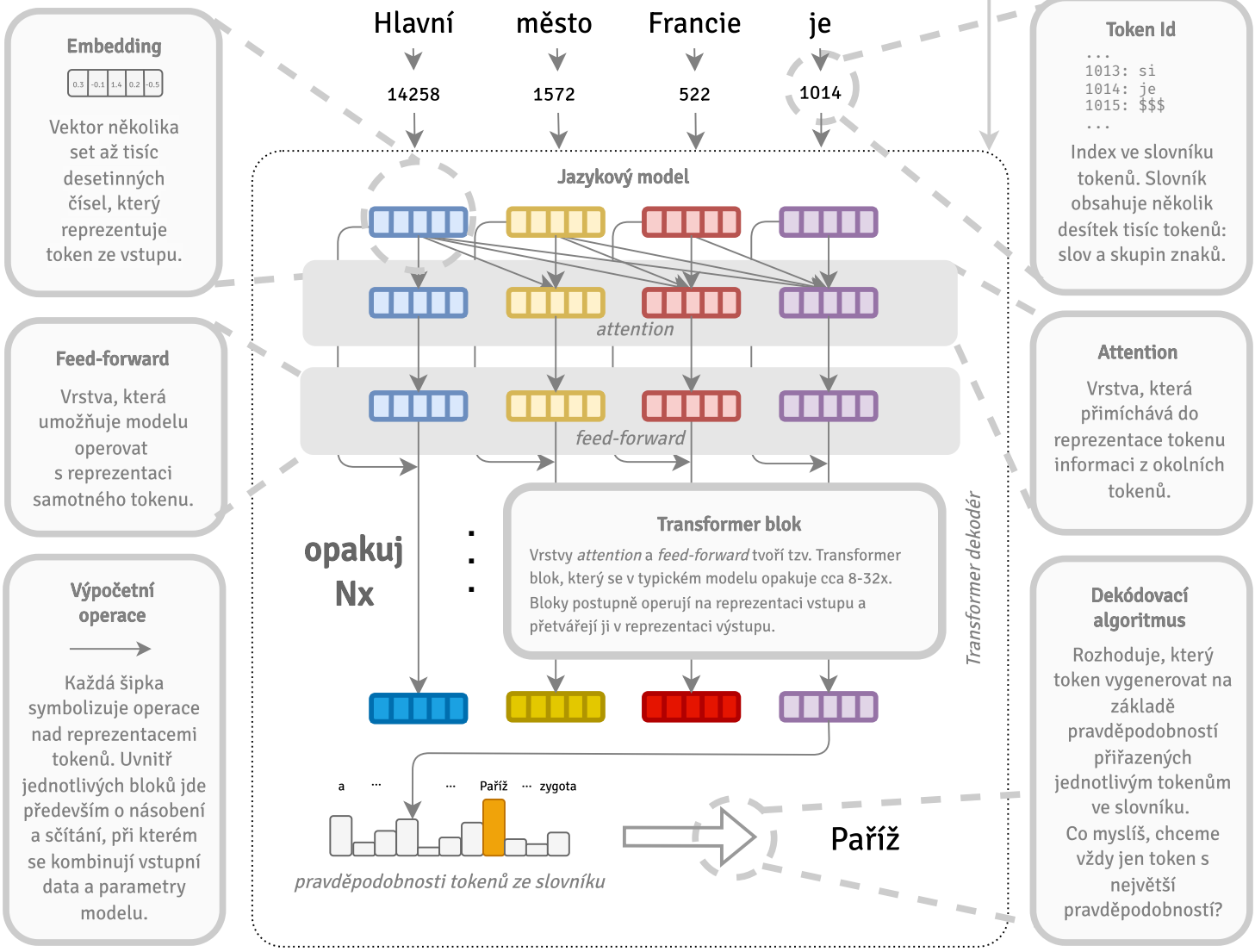
a) Dotrénuj ho na textech, ve kterých se plní instrukce.

[instrukce] Napiš definici slova "počítač". [odpověď] Počítač je elektronický přístroj, ...

b) Dotrénuj ho, aby preferoval odpovědi, které se lidem líbí.

uživatel: *Máš to špatně!*
odpověď #1: *No to teda nemám!*
odpověď #2: *Moje chyba, omlouvám se, ...*

Jeden krok generování textu



Embedding

Vektor několika set až tisíc desetinných čísel, který reprezentuje token ze vstupu.

Token Id

```
...
1013: si
1014: je
1015: $$$
...
```

Index ve slovníku tokenů. Slovník obsahuje několik desítek tisíc tokenů: slov a skupin znaků.

Feed-forward

Vrstva, která umožňuje modelu operovat s reprezentaci samotného tokenu.

Attention

Vrstva, která přimíchává do reprezentace tokenu informaci z okolních tokenů.

Výpočetní operace

Každá šipka symbolizuje operace nad reprezentacemi tokenů. Uvnitř jednotlivých bloků jde především o násobení a sčítání, při kterém se kombinují vstupní data a parametry modelu.

Dekódovací algoritmus

Rozhoduje, který token vygenerovat na základě pravděpodobností přiřazených jednotlivým tokenům ve slovníku. Co myslíš, chceme vždy jen token s největší pravděpodobností?

Velké jazykové otázky a odpovědi

Umí si jazykové modely i dohledávat odpovědi na internetu?

Krátká odpověď: Neumí.

Dlouhá odpověď: Jazykový model jako takový hledat na internetu neumí. Všechno, co umí, se musel naučit během trénování.

Služby jako např. Google Gemini ale mají externí vyhledávací modul. Tento modul hledá informace související s dotazem pomocí běžného vyhledávače a předkládá je jazykovému modelu na vstupu společně se vstupem od uživatele. Model pak s nimi pracuje stejně, jako se zbytkem textu. Kvalita těchto výsledků může ovlivnit vygenerovaný text jak pozitivně, tak negativně.

Má model seznam všech slov ve všech jazycích?

Krátká odpověď: Ne – ale jejich částí ano!

Dlouhá odpověď: Všech slov by bylo opravdu až příliš (už jen všechny ty vyskloňované tvary v češtině!). Proto používáme tzv. "subwordy": slova a části slov, ze kterých můžeme libovolná slova poskládat. Frekventovanější slova máme ve slovníku přímo, ta méně častá složíme z více částí.

Rozsekat text na subwordy a pak ho zase poskládat je úkolem specializovaného algoritmu, tzv. tokenizéru, který pracuje nezávisle na jazykovém modelu.

Proč modely neumí ostatní jazyky tak dobře, jako angličtinu?

Krátká odpověď: Protože angličtina je jazykem internetu.

Dlouhá odpověď: Zatím nedokážeme učit modely jazyk ani zdaleka tak efektivně, jako třeba děti. Ani celá Wikipedie nestačí na to, aby se velký jazykový model naučil modelovat jazyk bez chyb.

Schopnosti modelu proto rostou s množstvím textů, na kterých byl trénovaný. A textů v angličtině se dá najít opravdu hodně! Překvapivě i v češtině jich je celkem dost, ale třeba v irštině nebo telugu bude model pokulhávat.

Jak energeticky náročné je natrénovat model? A kolik energie spotřebuje jeden dotaz?

Krátká odpověď: Trénovat je náročné, generovat ne tolik.

Dlouhá odpověď: Záleží na velikosti modelu, ale odhaduje se, že natrénovat model s 175 mld. parametrů stojí 1.2 GWh energie, což odpovídá roční spotřebě 120 amerických domácností. Modely se ale naštěstí netrénují tak často.

Generovat z natrénovaného modelu text už je mnohem efektivnější: průměrná odpověď ChatGPT spotřebuje cca 3 Wh, což odpovídá např. rychlovarné konvici zapnuté na 10 sekund.

[1] https://googleblog.blogspot.com/2009/01/powering-google-search.html
[2] https://adasci.org/how-much-energy-do-llms-consume-unveiling-the-power-behind-ai/
[3] https://andymasley.substack.com/p/a-cheat-sheet-for-conversations-about

Co dělá velké jazykové modely tak inteligentní?

Krátká odpověď: To je mi ale těžká otázka...!

Dlouhá odpověď: Modely do nějaké míry umějí zopakovat to, co viděly v trénovacích datech. Chytrá odpověď proto mohla jen "ležet na internetu".

Modely ale umí částečně i generalizovat: kombinovat naučené vzory v originální odpovědi. Důvodem může být to, že s velkým množství dat může být generalizovat jednodušší, než se učit vše nazpaměť.

A tady už přicházejí další otázky: Co dělá inteligentní člověka? A co je to vlastně ta "inteligence"?

Kde se můžu naučit o jazykových modelech víc?

Krátká odpověď: U nás na ÚFALu!

Dlouhá odpověď: Podobnými tématy se zabýváme na Ústavu Formální a Aplikované Lingvistiky (ÚFAL), což je součást Matematicko-fyzikální fakulty Univerzity Karlovy.

Na naší katedře propojujeme znalosti o jazyce se znalostmi ze strojového učení. Učíme o tom bakalářské i magisterské předměty, jako třeba Deep Learning nebo Large Language Models.

vytvořil Zdeněk Kasner, MFF UK

https://kasnerz.github.io/assets/materials/llm_tahak.pdf