Západočeská univerzita v
Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

DIPLOMOVÁ PRÁCE

Porozumění řeči založené na neuronových sítích

Plzeň, 2021 Bc. Adam Frémund

ZÁPADOČESKÁ UNIVERZITA V PLZNI

Fakulta aplikovaných věd

Akademický rok: 2020/2021

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Adam FRÉMUND**
Osobní číslo:	**A18N0054P**
Studijní program:	**N3918 Aplikované vědy a informatika**
Studijní obor:	**Kybernetika a řídicí technika**
Téma práce:	**Porozumění řeči založené na neuronových sítích**
Zadávající katedra:	**Katedra kybernetiky**

Zásady pro vypracování

1. Nastudujte problematiku neuronových sítí v oblasti porozumění mluvené řeči.
2. Zaměřte se na techniky nazývané transfer-learning a jejich použití v porozumění mluvené řeči.
3. Vyberte popř. sestavte vhodnou experimentální datovou sadu a popište ji.
4. Pro datovou sadu navrhněte strukturu modelu, model natrénujte a vyhodnoťte.
5. Dosažené výsledky analyzujte a navrhněte případné vylepšení.

Rozsah diplomové práce: **40-50 stránek A4**
Rozsah grafických prací:
Forma zpracování diplomové práce: **tištěná**

Seznam doporučené literatury:

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, (Mlm). Retrieved from http://arxiv.org/abs/1810.04805
Liu, X., He, P., Chen, W., & Gao, J. (2019). Multi-Task Deep Neural Networks for Natural Language Understanding, 4487–4496. https://doi.org/10.18653/v1/p19-1441
Lee, S., & Jha, R. (2019). Zero-Shot Adaptive Transfer for Conversational Language Understanding. Proceedings of the AAAI Conference on Artificial Intelligence, 33, 6642-6649. https://doi.org/10.1609/aaai.v33i01.33016642
Další literaturu dodá vedoucí DP.

Vedoucí diplomové práce: **Ing. Jan Švec, Ph.D.**
Katedra kybernetiky

Datum zadání diplomové práce: **1. října 2020**
Termín odevzdání diplomové práce: **24. května 2021**

Západočeská univerzita v Plzni
Fakulta aplikovaných věd

L.S.

Doc. Dr. Ing. Vlasta Radová
děkanka

Prof. Ing. Josef Psutka, CSc.
vedoucí katedry

V Plzni dne 1. října 2020

PROHLÁŠENÍ

Předkládám tímto k posouzení a obhajobě diplomovou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Tlučné dne .....

Poděkování

Tímto bych chtěl poděkovat panu Ing. Janu Švecovi, Ph.D. za skvělé vedení práce a ochotu při poskytování cenných rad a věcných připomínek, které vedly k úspěšnému vypracování této práce i v současné nelehké době.

Abstrakt

Tato diplomová práce se zabývá použitím neuronových sítí pro zpracování přirozeného jazyka. V posledních letech se v této oblasti umělé inteligence začaly hojně využívat modely s architekturou *Transformer*. Právě tyto modely a jejich struktura jsou důkladně rozebrány v teoretické části této práce. Bližší pohled je pak věnován jedněm z nejmodernějších a nejúspěšnějších modelů, a to modelům typu BERT a T5.

V praktické části je s těmito modely experimentováno. Jsou použity pro řešení dvou úloh zpracování přirozeného jazyka. První úlohou je analýza sentimentu anglických a českých filmových recenzí. Druhou úlohou je detekce entit. Nejprve jsou modely použity pro detekci pojmenovaných entit v textech. Následně je T5 model aplikován pro porozumění mluvené řeči, především pak k detekci sémantických entit v řečových korpusech. Tato diplomová práce je prvním vědeckým článkem věnujícím se T5 modelu pro úlohy zpracování českého jazyka.

Klíčová slova

zpracování přirozeného jazyka, porozumění řeči, neuronové sítě, transformer achitektura, BERT model, T5 model, analýza sentimentu, detekce pojmenovaných entit

Abstract

This master thesis deals with the use of artificial neural networks for natural language processing. In recent years, models with the *Transformer* architecture have become widely used in this field of artificial intelligence. It is these models and their structure that are thoroughly discussed in the theoretical part of this work. A closer look is devoted to BERT and T5 models.

In the practical part, these models are experimented with. They are used to solve two natural language processing tasks. Firstly, to analyze the sentiment of English and Czech film reviews. The second task is an entity detection. First, models are used to detect named entities in texts. And then the T5 model is applied for the spoken language understanding task, especially for the detection of semantic entities in speech corpora. This master thesis is the first scientific article devoted to the T5 model for Czech language processing tasks.

Key words

natural language processing, speech comprehension, artificial neural networks, transformer architecture, BERT model, T5 model, sentiment analysis, named entity recognition

Obsah

1 Úvod **1**

2 Transformer architektura **2**
2.1 Úvod do neuronových sítí . . . 2
2.2 Neuronové sítě v oblasti zpracování řeči a přirozeného jazyka . . . 3
2.3 Architektura transformer modelu . . . 4
2.3.1 Enkodér a Dekodér bloky . . . 4
2.3.1.1 Enkodér . . . 5
2.3.1.2 Dekodér . . . 5
2.3.2 Attention . . . 5
2.3.2.1 Scaled Dot-Product Attention . . . 6
2.3.2.2 Multi-Head attention . . . 8
2.3.3 Dopředné sítě . . . 8
2.4 Vektorová reprezentace slov . . . 8
2.4.1 One-hot vektor reprezentace . . . 8
2.4.2 Vnoření slov (Word Embeddings) . . . 9
2.4.2.1 Word2Vec . . . 9
2.5 Vektorová reprezentace textu . . . 11
2.5.1 Bidirectional Encoder Representations from Transformer (BERT) . . . 11
2.5.1.1 Architektura . . . 12
2.5.1.2 Předzpracování textu . . . 13
2.5.1.3 Předtrénování modelu . . . 14
2.5.1.4 Fine-Tuning . . . 15
2.6 Model T5 . . . 15
2.6.1 Text-to-Text Framework . . . 15
2.6.2 Architektura modelu . . . 16
2.6.3 Předtrénování modelu . . . 17
2.6.4 Textový korpus pro předtrénování modelu . . . 18
2.6.5 Použití modelu . . . 19

3 Vývojové prostředí **20**
3.1 Framework . . . 20
3.1.1 TensorFlow . . . 20
3.1.1.1 Principy . . . 20

3.1.2 PyTorch . . . 22
3.1.2.1 Principy . . . 22
3.2 Výpočetní prostředí . . . 22

4 Experimenty **23**
4.1 Analýza sentimentu . . . 23
4.1.1 Dataset IMDB recenzí . . . 23
4.1.1.1 Trénovací data . . . 23
4.1.1.2 BERT model . . . 24
4.1.1.3 T5 model . . . 25
4.1.2 Dataset ČSFD recenzí . . . 25
4.1.2.1 Trénovací data . . . 25
4.1.2.2 BERT model . . . 25
4.1.2.3 T5 model . . . 26
4.1.3 Výsledky . . . 26
4.1.3.1 Evaluační metriky . . . 27
4.1.3.2 Analýza sentimentu IMDB recenzí . . . 29
4.1.3.3 Analýza sentimentu ČSFD recenzí . . . 31
4.2 Detekce pojmenovaných entit . . . 34
4.2.1 Dataset CNEC . . . 35
4.2.1.1 Trénovací data . . . 35
4.2.1.2 BERT model . . . 37
4.2.1.3 T5 model . . . 39
4.2.2 Dataset HHTT + TIA . . . 40
4.2.2.1 Trénovací data . . . 40
4.2.2.2 T5 model . . . 43
4.2.3 Výsledky . . . 44
4.2.3.1 Evaluační metriky . . . 44
4.2.3.2 Detekce pojmenovaných entit pro CNEC textový korpus . . . 45
4.2.3.3 Detekce sémantických entit pro HHTT + TIA korpus 51
4.3 Zhodnocení výsledků . . . 54

5 Závěr **55**

Literatura **57**

Souhrn dosažených výsledků **61**

Seznam obrázků

2.1 Dopředná neuronová síť 2
2.2 Transformer architektura 5
2.3 Self-attention mechanismus 6
2.4 Tvorba Query, Key a Value vektorů ze vstupního slova 7
2.5 Multi-Head Attention 7
2.6 Ukázka One-hot reprezentace 9
2.7 Ukázka kontextu slova v textu 10
2.8 Architektura CBOW a Skip-gram modelu 10
2.9 Vektorové operace 11
2.10 Trénování BERT modelu 12
2.11 Vstupní vektory 13
2.12 Princip T5 modelu 16
2.13 Jednotlivé architektury modelů 16
2.14 Vývojový diagram trénování T5 modelu 18
2.15 Použití T5 modelu 19

3.1 Výpočetní graf jednoduchých matematických operací 21

4.1 Chybová matice pro vyhodnocení úlohy binární klasifikace 27
4.2 Přesnost a úplnost 28
4.3 Vliv délky vstupní sekvence - IMDB 29
4.4 Srovnání výsledků BERT a T5 modelu - IMDB 30
4.5 Chybové matice - IMDB 30
4.6 Vliv délky vstupní sekvence - ČSFD 32
4.7 Srovnání výsledků BERT a T5 modelu - ČSFD 32
4.8 Chybové matice - ČSFD 33
4.9 Výstup BERT modelu - 7 entit 38
4.10 Výstup BERT modelu - 42 entit 39
4.11 Srovnání BERT a T5 modelu pro CNEC korpus - 7 entit 47
4.12 Srovnání BERT a T5 modelu pro CNEC korpus - 42 entit 48
4.13 Chybové matice - 7 entit 49
4.14 Vliv trénovacích dat na výsledné hodnoty 52

Seznam tabulek

2.1 Přístupy pro předtrénování T5 modelu 17

4.1 Počet dostupných trénovacích a testovacích dat - IMDB 24
4.2 Počet dostupných trénovacích a testovacích dat - ČSFD 25
4.3 Nejlepší dosažené výsledky - IMDB 31
4.4 Nejlepší dosažené výsledky - ČSFD 34
4.5 Entity korpusu CNEC 36
4.6 Ukázka anotovaného dialogu 41
4.7 Sémantické koncepty korpusu HHTT 41
4.8 Ukázkové promluvy z korpusu TIA 42
4.9 Sémantické entity, sémantické akce a sémantické cíle korpusu TIA. . . 43
4.10 Počty vět v jednotlivých tsv souborech datasetu HHTT + TIA 44
4.11 Nejlepší dosažené výsledky - CNEC 7 entit 47
4.12 Nejlepší dosažené výsledky - CNEC 42 entit 48
4.13 Průměrné hodnoty F-míry jednotlivých entit 50
4.14 Průměrné hodnoty F1-míry, přesnosti a úplnosti pro detekci entit. . . 51
4.15 Průměrné hodnoty přesnosti T5 modelu 52
4.16 Nejlepší dosažené hodnoty přesnosti pro úlohu detekce sémantických entit 53

1 Úvod

Zpracování řeči a přirozeného jazyka (NLP) je oblastí umělé inteligence. Jedná se o obor, jehož cílem je naučit stroje zpracovávat jazyk tak, jako to umí člověk. Jinými slovy řečeno jde o schopnost jazykovému projevu porozumět a také sdělení v přirozeném jazyce vyprodukovat. V dnešní době, kdy roste popularita hlasových asistentů (Amazon Alexa, Google Home, atd.), se zpracování přirozeného jazyka stává velice perspektivní disciplínou. Mezi typické úlohy zpracování přirozeného jazyka patří převod řeči na text, syntéza řeči, analýza sentimentu, detekce pojmenovaných entit a mnoho dalšího. Poslední dvě zmíněné úlohy jsou předmětem experimentů této diplomové práce.

Pro strojové zpracování přirozeného jazyka se v posledním desetiletí začaly hojně používat neuronové sítě. Jedním z nejmodernějších přístupů je tzv. *Transformer* architektura. Teoretická část této diplomové práce je zaměřena na popis jednotlivých komponent těchto modelů. Především se věnuje modelu typu BERT, který dosáhl mnoha state-of-the-art výsledků v úlohách zpracování přirozeného jazyka. Druhým zkoumaným modelem je model s architekturou T5 (Text-To-Text Transfer Transformer). Jedná se o zcela nový přístup, jež umožňuje použití jednoho modelu pro několik specifických úloh zpracování přirozeného jazyka, aniž by bylo potřeba měnit jeho architekturu. V závěru teoretické části jsou popsány frameworky použité pro snadnější práci s neuronovými sítěmi.

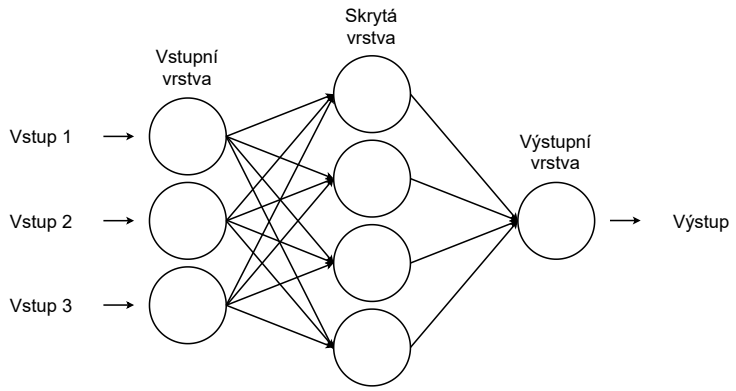
Tématem této diplomové práce je porozumění řeči založené na neuronových sítích. Modely používané pro porozumění řeči bývají shodné s těmi, které se používají pro zpracování přirozeného jazyka v textové podobě. Proto jsou modely, popsané v teoretické části, nejprve použity pro několik úloh porozumění textu. Prvním cílem je provést srovnání obou modelů, tj. BERT a T5 modelu. V první řadě jsou obě architektury použity pro analýzu sentimentu anglických a českých recenzí z internetových filmových databází. Druhou úlohou, na které jsou porovnány výkonnosti obou modelů, je detekce pojmenovaných entit. Poslední zkoumaná úloha je již z oblasti porozumění řeči. Jedná se o detekci sémantických entit v řečovém korpusu. Tato úloha ukazuje použití T5 modelu s nedokonalým výstupem systému automatického rozpoznávání řeči. Jedná se o klíčovou úlohu při implementaci zmíněných hlasových asistentů. Druhým cílem této diplomové práce je zjistit, zdali je T5 model v úloze detekce sémantických entit konkurence schopný pro model navržený speciálně pro danou úlohu.

2 Transformer architektura

Následující část je věnována architektuře typu *Transformer*, která se v posledních letech stala velice oblíbenou v oblasti umělé inteligence, především pak ve zpracování řeči a přirozeného jazyka.

2.1 Úvod do neuronových sítí

Umělé neuronové sítě jsou inspirovány lidskou nervovou soustavou. Přirozené i umělé neuronové sítě mají stejný základní prvek, a to neuron. Tyto neurony jsou navzájem propojeny a jsou schopny si mezi sebou předávat informace. Každá neuronová síť je mimo jiné charakterizována typem neuronů, jejich uspořádáním a strategií zvolenou při trénování takové sítě. Na obrázku 2.1 je znázorněn princip nejčastěji používané dopředné neuronové sítě (dalšími typy jsou například rekurentní a long short-term memory neuronové sítě). V takové síti se informace šíří jednosměrně od vstupu k výstupu. Je vidět, že jednotlivé neurony jsou uspořádány do vrstev. První vrstva je nazývána vstupní vrstvou a poslední vrstva je pojmenována jako vrstva výstupní. Vnitřní vrstvy jsou souhrnně označeny jako skryté.



Obrázek 2.1: Uspořádání neuronů v dopředné neuronové síti.

V každém neuronu probíhají dva procesy. Nejprve dojde k výpočtu potenciálu podle následujícího vzorce:

$$\xi = \sum_{i=1}^{n} w_i x_i - \theta, \tag{2.1}$$

kde w_i jsou jednotlivé váhové koeficienty vazeb neuronů, θ jsou prahové hodnoty a x_i jsou jednotlivé vstupní hodnoty. Následně je vypočtena hodnota výstupu pomocí tzv. aktivační funkce - například sigmoidální:

$$y = f(\xi) = \frac{1}{1 + \exp^{-\lambda\xi}} \tag{2.2}$$

Ze vzorců 2.1 a 2.2 si lze všimnout, že výstupní hodnoty neuronů jsou ovlivněny hodnotami jednotlivých váhových a prahových koeficientů. Tyto hodnoty jsou upraveny při procesu trénování.

Existují tři přístupy v trénování umělých neuronových sítí - s informací od učitele, bez informace od učitele a tzv. self-supervised trénování. Při trénování s informací od učitele jsou dostupné vstupní hodnoty a k nim jsou přiřazeny očekávané hodnoty na výstupu. Vstupní hodnoty se nechají propagovat neuronovou sítí, dojde k porovnání výstupu neuronové sítě s očekávanými hodnotami od učitele a následně jsou upraveny hodnoty vah a prahů tak, aby rozdíl mezi skutečným a očekávaným výstupem byl co nejmenší. Pokud je síť dobře nastavena a zároveň je při procesu trénování použita reprezentativní množina vstupů a výstupů, je pak taková síť schopna řešit i úlohy, se kterými se během učení přímo nesetkala.

V případě trénování bez informace od učitele jsou známy pouze vstupní hodnoty. Sítě, u kterých se používá tento princip trénování, se většinou věnují tzv. shlukové analýze (roztřídění množiny trénovacích dat do konečného množství tříd). Vstupní hodnoty jedné třídy jsou si vzájemně podobné.

Self-supervised trénování se často využívá právě u tzv. *Transfomer* modelů (viz. sekce 2.5.1.3 a 2.6.3), kterým je věnována následující část.

2.2 Neuronové sítě v oblasti zpracování řeči a přirozeného jazyka

Rekurentní neuronové sítě, long short-term memory (LSTM) neuronové sítě a tzv. gated recurrent units (GRU) byly dlouhou dobu brány jako state-of-the-art modely pro tvorbu jazykového modelu a především pak pro strojový překlad. Avšak v roce 2017 byl v článku *Attention Is All You Need* [1], na kterém se podíleli autoři pracující v Googlu, představen zcela nový přístup k řešení těchto úloh. Transformer architektura se od té doby stala standardem při řešení úloh zpracování přirozeného jazyka. Tato architektura má však své uplatnění například i v oblastech počítačového vidění.

Rekurentní neuronové sítě s využitím GRU nebo LSTM zpracovávají vstupní tokeny (token je jednotka, která většinou odpovídá jednotlivým slovům ve vstupní sekvenci) postupně a udržují stav, jež popisuje dosud viděná data. Tento stav je aktualizován po každém příchozím tokenu. Tudíž při zpracování *n-tého* tokenu je modelem kombinován stav popisující sekvenci *n-1* tokenů s informací o novém tokenu. Touto kombinací je vytvořen stav reprezentující sekvenci n tokenů. Teoreticky

je informace o předchozích tokenech schopná sama sebe propagovat skrz celou sekvenci tokenů. Tzn. stav vygenerovaný po příchodu *n-tého* tokenu na vstup obsahuje informaci o všech předchozích tokenech. V praxi se však tento postup ukazuje jako nedokonalý. Především kvůli tzv. problému mizejícího gradientu neobsahuje stav na konci dlouhé sekvence tokenů přesnou a extrahovatelnou informaci o předešlých tokenech.

Tento problém byl vyřešen zavedením tzv. attention mechanismů. Samotné anglické slovo *"Attention"* znamená zaměření se na určitou část něčeho a bližší věnování se této části. Attention mechanismus tento koncept také využívá a zaměřuje se na určité faktory při zpracování vstupních dat. Attention vrstva může přistupovat ke všem předchozím stavům. Tyto stavy váží pomocí natrénovaných měr relevance pro aktuální token, a poskytuje tak mnohem přesnější informaci o vztahu mezi vzdálenějšími tokeny. Tento mechanismus zvýšil výkon rekurentních neuronových sítí. Avšak s příchodem transformer konceptu vyšlo najevo, že není potřeba rekurentního zpracování pro získání stejných nebo lepších výsledků. Transformer zpracovává všechny tokeny v jednu chvíli a počítá attention váhy mezi všemi tokeny. Narozdíl od sekvenčního zpracování tokenů tento přístup umožňuje efektivní trénování i na velkých datasetech. [2][3]

2.3 Architektura transformer modelu

Jak bylo v úvodu řečeno, transformer architektura byla poprvé představena v článku *Attention Is All You Need* [1]. Podobně jako předešlé používané modely se jedná o architekturu enkodér-dekodér. Enkodér mapuje sekvenci vstupních symbolů $(x_1, ..., x_n)$ na sekvenci $\mathbf{z} = (z_1, ...z_n)$. Ze sekvence $\mathbf{z}$ poté dekodér generuje sekvenci výstupních symbolů. Tento model je auto regresivní a v každém kroku se na vstupu objevují již vygenerované výstupní symboly.

V článku navržený transformer model obsahuje za sebou poskládané bloky obsahující tzv. self-attention blok a fully-connected dopřednou neuronovou síť - a to jak pro blok enkodéru, tak i pro dekodér, viz. obrázek 2.2. [1]

2.3.1 Enkodér a Dekodér bloky

Enkodér a dekodér jsou bloky používané v modelech využívajících transformaci sekvence na sekvenci. Zároveň jsou nedílnou součástí transformer architektury. Obecně je enkodér blok neuronová síť, která vstupní sekvenci přemění na tzv. vektor příznaků (jinými slovy zakóduje vstupní sekvenci). Dekodér blok má vstupem zmíněný vektor příznaků a snaží se ho přeměnit na odpovídající sekvenci[1].

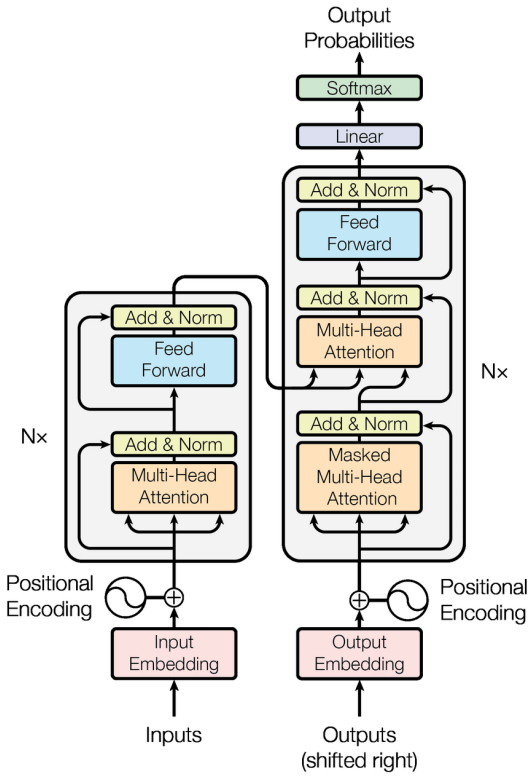
[1]Například strojový překlad, při kterém je cílem převést sekvenci v jednom jazyce na odpovídající sekvenci v jazyce druhém.

2.3.1.1 Enkodér

V původním článku je enkodér složen ze 6 identických vrstev. Každá vrstva má dvě dílčí vrstvy. První dílčí vrstvou je tzv. multi-head self-attention mechanismus. Druhou vrstvou je obyčejná dopředná plně propojená neuronová síť. Jak je vidět na obrázku 2.2, výstup každé z dílčích vrstev je normalizován jako Norm(x + SubLayer(x)), kde SubLayer(x) je funkce implementována samotnou dílčí vrstvou. Výstupy všech dílčích vrstev produkují výstup o dimenzionalitě d_{model} (v praxi například $d_{model} = 512$). [1]

2.3.1.2 Dekodér

Podobně jako enkodér blok je i dekodér blok složen ze 6 identických vrstev. Navíc však dekodér oproti enkodér bloku obsahuje dvě vrstvy multi-head self-attention. Vstupem této přidané sub vrstvy je výstup z enkodér bloku. Zároveň je původní struktura attention vrstvy pozměněna, aby se předešlo přístupu k následujícím datům, tzn. odhad pro pozici i závisí pouze na známých výstupech na pozici menší než i. [1]

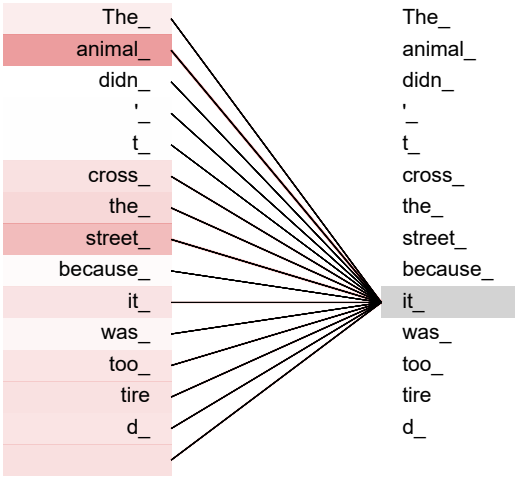


Obrázek 2.2: Transformer architektura. [1]

2.3.2 Attention

Funkce Attention vrstvy může být popsána jako matematická operace nad tro-

jicí vektorů query q, key k a value v. Výstup je dán váženou sumou vektorů v, kde váha přiřazena každému vektoru v je spočtena skalárním součinem vektoru q a odpovídajícího vektoru k. Cílem těchto vrstev je zachytit vazby mezi jednotlivými vstupními tokeny. [1] Na obrázku 2.3 je graficky vizualizován mechanismus self-attention vrstvy. Je patrné, že slovo *it* (český překlad: to) má největší vazbu ke slovu *animal* (český překlad: zvíře).

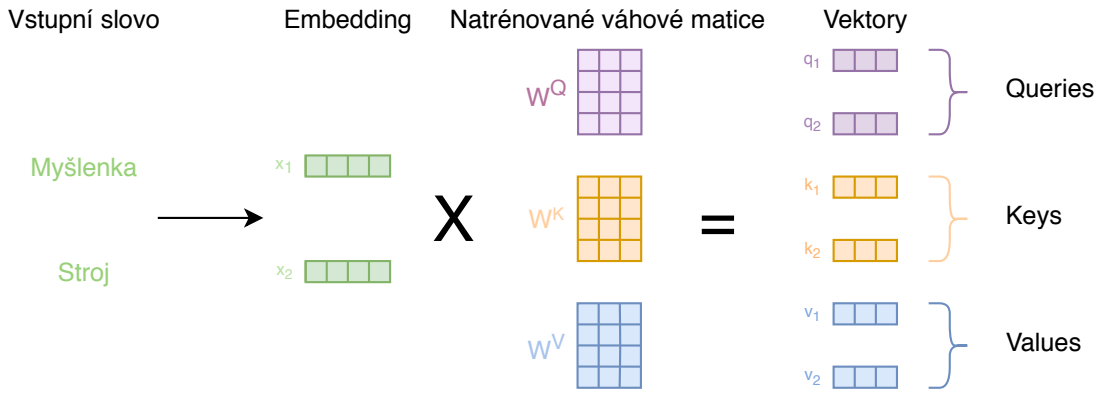


Obrázek 2.3: Vizualizace self-attention mechanismu. Tento příklad popisuje vazby slova *it* (v češtině *to*) k ostatním slovům ve větě. Data získána z jedné hlavy poslední multi-head attention vrstvy BERT modelu.[2]

2.3.2.1 Scaled Dot-Product Attention

Prvním krokem při výpočtu self-attention je vytvoření tří vektorů z každého vstupního vektoru enkodéru. Pro každé slovo je vytvořen Query vektor q, Key vektor k a Value vektor v. Tyto vektory jsou získány vynásobením embedding vektoru (viz. část 2.5.1.2) třemi váhovými maticemi W^Q, W^K, W^V, které jsou natrénovány v průběhu trénovacího procesu. [1]

[2]Příklad vytvořen v Google-Colab [4].

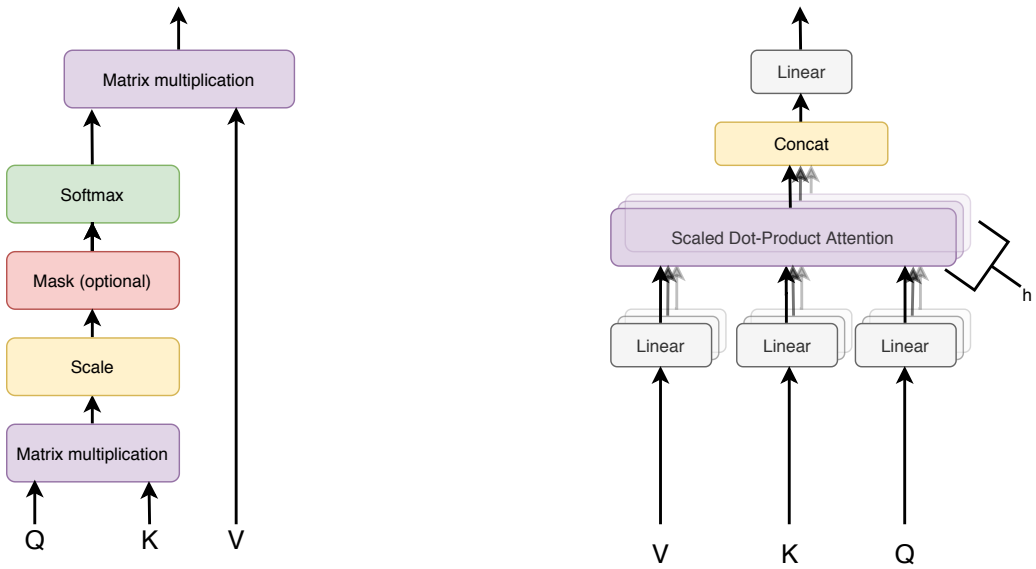


Obrázek 2.4: Tvorba Query, Key a Value vektorů ze vstupního slova. [5]

Výpočet tzv. "Scaled Dot-Product Attention" byl navržen následovně:

$$Attention(Q, K, V) = Softmax(\frac{Q \cdot K^T}{\sqrt{d_k}}) \cdot V, \tag{2.3}$$

kde matice Q, K, V jsou matice složené z jednotlivých query, key a value vektorů. Vstupní embedding vektory jsou řádky matice X. Potřebné matice Q, K, V vzniknou přenásobením matice X příslušnými váhovými maticemi W^Q, W^K, W^V. Při předpokladu, že matice Q, K jsou obecně d_k-dimenzionální a jejich komponenty jsou nezávislé proměnné se střední hodnotou 0 a variancí 1, poté jejich skalární součin má střední hodnotu 0 a varianci d_k. Při použití *Softmax* je důležité, aby variance byla 1, tudíž je zmíněný skalární součin matic Q, K normován $\sqrt{d_k}$. Práce s maticemi umožňuje rychlejší zpracování vstupních embedding vektorů. Grafické znázornění výpočtu scaled dot-product attention je na obrázku 2.5 v levé části. [1]



Obrázek 2.5: (vlevo) Postup výpočtu Scaled Dot-Product Attention. (vpravo) Multi-Head Attention složený hned z několika attention vrstev zpracovávajících data paralelně. [1]

2.3.2.2 Multi-Head attention

Článek [1] zároveň přinesl návrh na vylepšení attention vrstvy. Vrstva je rozšířena o tzv. "Multi-Headed" attention. Tento modul je schopný počítat hned několik oddělených attention mechanismů najednou. Výstupy jednotlivých attention vrstev jsou následně konkatenovány a lineárně transformovány do očekávané výstupní dimenze. Multi-head attention modul umožňuje odlišné přístupy k vstupní sekvenci, tzn. je schopný rozlišovat různé závislosti jednotlivých vstupních tokenů. [6]

$$\begin{aligned} MultiHead(Q,K,V) &= Concat(head_1,...,head_h)W^O \\ \text{kde} \qquad head_i &= Attention(QW_i^Q, KW_i^K, VW_i^V). \end{aligned} \tag{2.4}$$

Matice W_i^Q, W_i^K, W_i^V, W^O v rovnici 2.4 jsou projekční matice, jejichž hodnoty jsou učeny při trénování. V článku navržený model obsahuje $h = 8$ paralelních attention vrstev. Jednotlivé "hlavy" Multi-Head attention mechanismu mají sníženou dimenzi a následná výpočetní náročnost model je srovnatelná s modelem, jež obsahuje pouze Single-Head attention mechanismus s plnou dimenzí. [1]

2.3.3 Dopředné sítě

Každá vrstva enkodéru a dekodéru kromě attention vrstvy obsahuje plně propojenou dopřednou neuronovou síť. Tato neuronová síť je použita identicky na každou pozici výstupu attention vrstvy. Vstupní i výstupní dimenze těchto neuronových sítí d_{model} je opět závislá na typu modelu (v praxi například $d_{model} = 512$). [1]

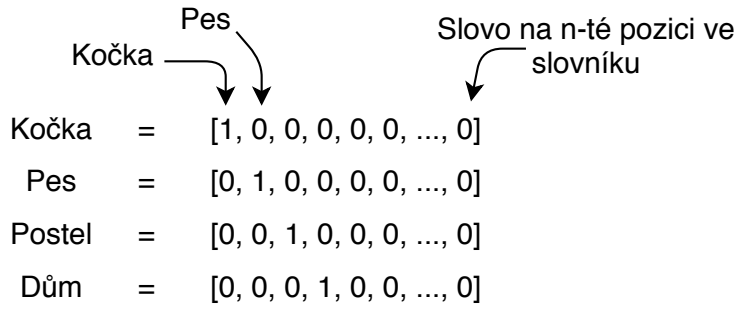
2.4 Vektorová reprezentace slov

Při úlohách strojového zpracování přirozeného jazyka je důležité převést text, se kterým je operováno, do reprezentace, která vyhovuje i výpočetním zařízení, tzn. převést text do číselné podoby. Je hned několik přístupů jak tento převod provést, například tzv. One-hot vektor reprezentace, Bag-of-words přístup, N-gramové modely, TF-IDF modely a tzv. Vnoření slov (angl. Word Embedding). Tato část bude věnována především One-hot vektor a Word Embedding přístupu.

2.4.1 One-hot vektor reprezentace

Jedna z nejjednodušších forem reprezentace slov je tzv. One-hot vektor reprezentace. Zjednodušeně řečeno je slovo reprezentováno jediným vektorem dimenze N, který obsahuje $N - 1$ prvků s hodnotou 0 a jeden prvek s hodnotou 1 (tzv. hot pozice). Ukázka možné One-hot reprezentace je na obrázku 2.6.

Dimenze N je dána velikostí slovníku slov, jež je pro danou úlohu používán. Každá pozice ve vektoru reprezentuje jednotlivé slovo ve slovníku a z toho plyne, že hodnota 1 je na pozici vektoru, který odpovídá indexu slova v použitém slovníku. Tento přístup je velice intuitivní a byl používán především v začátcích strojového zpracování přirozeného jazyka. Tvorba této reprezentace není výpočetně náročná a je velice jednoduchá na implementaci. Na druhou stranu však využití one-hot vektorů neposkytuje žádnou další informaci o slovech. To znamená, že vektory neobsahují informaci o vztazích mezi jednotlivými slovy a především neobsahují jejich kontext. Zároveň se tato reprezentace stává nevýhodnou při použití velkých slovníků. Dalším problémem je chybějící reprezentace slov nevyskytujících se ve slovníku, tj. neznámých slov. Tento nedostatek vedl k použití tzv. SentencePiece. Jedná se o jazykově nezávislý tokenizér, jež je představen v článku *SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing* [7]. [8]



Obrázek 2.6: Ukázka One-hot reprezentace.

2.4.2 Vnoření slov (Word Embeddings)

Zmíněná One-hot reprezentace využívá tzv. řídké vektory. Jedná se o vektory, které obsahují pouze malé množství nenulových prvků. Zároveň tyto vektory postrádají informaci o vztazích mezi slovy, jako je například kontextová závislost. Jako příklad lze úvest, že one-hot vektory nejsou schopné zachytit skutečnost, že slova *pes* a *kočka* jsou většinou používána pro označení domácích zvířat.

Naproti tomu word embedding vektory reprezentují slova jako vícedimenzionální vektory s čísly s plovoucí desetinnou čárkou. Jedná se o nízko-dimenzionální projekci z prostoru řídkých one-hot vektorů. Výhodou těchto vektorů je skutečnost, že sémanticky podobná slova jsou v N-dimenzionálním prostoru umístěna blízko sebe. To znamená, že slova kolo a motor by se měla nacházet blíže ke slovu auto (kvůli podobnosti významu slov) než například slovo jablko. [9]

2.4.2.1 Word2Vec

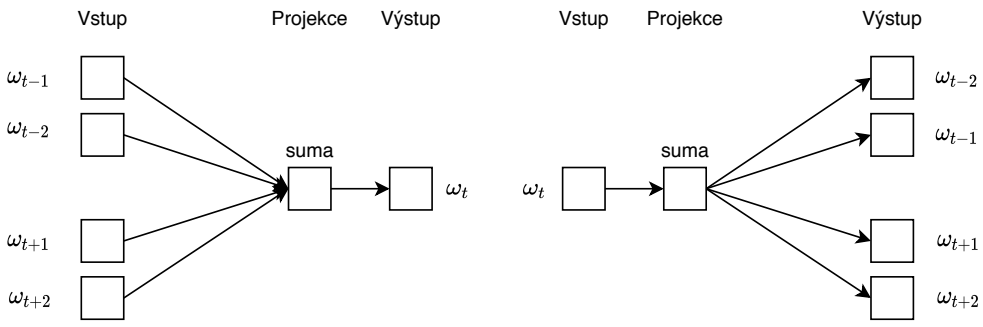
Word2Vec je technika/skupina modelů, jež generuje vektorovou reprezentaci slov za použití dvouvrstvých neuronových sítí. Díky Word2Vec je možné natrénovat

vektorovou reprezentaci i z obsáhlých trénovacích korpusů. Zahrnuje velké množství syntaktických a sémantických slovních vazeb. Vektorová reprezentace získaná právě zmíněným Word2Vec přístupem se ukázala být dobrým řešením pro nespočet úloh strojového zpracování přirozeného jazyka. Za autora této metody je považován Ing. Tomáš Mikolov Ph.D. a tým vědců z Google Inc., kteří tento přístup představili ve svém článku *Efficient Estimation of Word Representations in Vector Space* (2013) [10].

Význam slova je dán kontextem, ve kterém se slovo vyskytuje, a to se také stalo hlavní myšlenkou této techniky. Word2Vec využívá 2 okolních slov vždy po jednom z každé strany středového slova, viz. obrázek 2.7. Pro tvorbu vektorové reprezentace jsou použity dvě základní neuronové sítě. Continuous Bag of Words(CBOW) a Skip-Gram model. Vstupem obou těchto sítí jsou slova zakodovaná one-hot vektor reprezentací. CBOW na základě kontextu predikuje středové slovo a Skip-gram model naopak predikuje kontext na základě středového slova. Architektura těchto sítí je znázorněna na obrázku 2.8. Oba modely lze použít pro získání vektorové reprezentace slov. Ve zmíněném článku [10] se autoři zmiňují o výhodách obou modelů. Skip-gram model je vhodný použít, pokud je zapotřebí zpracovávat objemné textové korpusy, ale zároveň je mnohem pomalejší, co se týče trénování. Velikost kontextového okénka je obvykle volena 5 a dimenze výsledných vektorů je 300. [11]

Vstupní text	Kontext	Středové slovo
Včerejší závod vyhrál žokej ...	závod	Včerejší
Včerejší závod vyhrál žokej ...	Včerejší, vyhrál	závod
Včerejší závod vyhrál žokej ...	závod, žokej	vyhrál

Obrázek 2.7: Ukázka kontextu slova v textu.



Obrázek 2.8: Architektura CBOW (vlevo) a Skip-gram modelu (vpravo). [12]

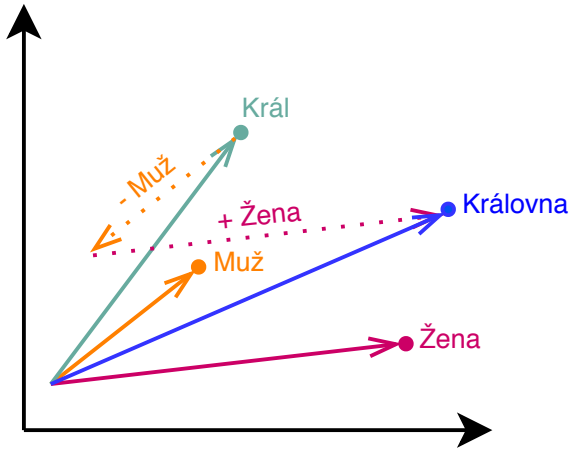
Vektorová reprezentace slova získaná metodou Word2Vec skrývá zajímavé matematické vztahy mezi jednotlivými slovy. Tyto vztahy lze velice dobře graficky

znázornit. Na obrázku 2.9 je vidět příklad imaginární rovnice

$$v_{\text{král}} - v_{\text{muž}} + v_{\text{žena}} \approx v_{\text{královna}}, \tag{2.5}$$

kde v je vektor odpovídající vektorové reprezentaci jednotlivých slov.

Word2Vec vektorová reprezentace má zároveň podobnou strukturu, i když je trénovaná na korpusech různých jazyků, tato vlastnost je velice užitečná pro strojový překlad. Tyto uvedené příklady však nejsou jedinou doménou word2vec vektorových reprezentací. Dále je lze použít pro detekci plural-singular vztahů mezi slovy, detekci genderových vztahů mezi slovy, zařazení slov textového korpusu do tříd a v mnoha dalších případech. [13]



Obrázek 2.9: Vektorové operace.

2.5 Vektorová reprezentace textu

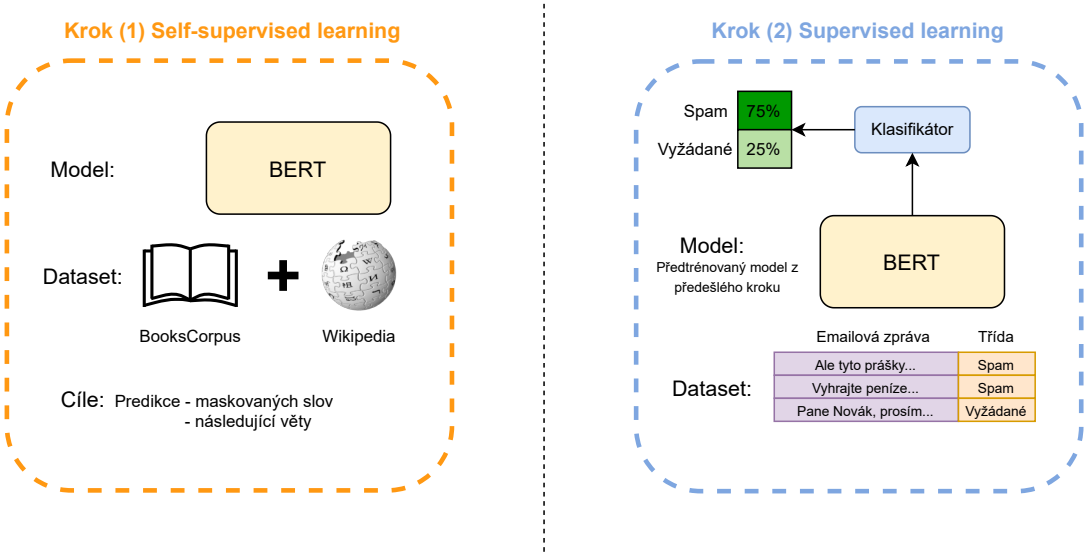
Word embeddings přístupy (Word2Vec, GloVe) poskytují pouze informaci o podobnosti jednotlivých slov. Pro zpracování přirozeného jazyka je však zapotřebí získat reprezentaci zachycující kontext jednotlivých slov ve větách. Natrénování modelu, který poskytne kontextovou reprezentaci slov se věnovali autoři článku *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding* [14] a přišli s řešením, tzv. Bidirectional Encoder Representations from Transformer zkráceně BERT modelem.

2.5.1 Bidirectional Encoder Representations from Transformer (BERT)

BERT je model vycházející z architektury Transformer. Transformer architektura je složena ze dvou oddělených mechanismů - enkodéru, který čte vstupní text

a dekodéru, jež produkuje predikci pro danou úlohu. BERT model využívá pouze mechanismus enkodéru.

Oproti směrovým modelům, které čtou vstupní text sekvenčně (zleva-doprava nebo zprava-doleva), BERT model čte celou vstupní sekvenci najednou. To modelu poskytuje znalost kontextu slova na základě okolních slov. [15] Této vlastnosti se využívá pro předtrénování modelu na textovém korpusu bez informace od učitele. K takto předtrénovanému modelu lze jednoduše přidat výstupní vrstvu a dotrénovat model na specifickou úlohu zpracování přirozeného jazyka - například klasifikace textu viz. obrázek 2.10.



Obrázek 2.10: Trénování BERT modelu. (1) Předtrénování: trénováno na velkém textovém korpusu (knihy a Wikipedia). (2) Fine-tuning: dotrénování modelu pro specifickou úlohu pomocí dat od učitele. [16]

2.5.1.1 Architektura

Architektura BERT modelu odpovídá vícevrstvému obousměrnému Transformer enkodéru [1] popsanému v části 2.3. Jinými slovy se BERT model skládá ze zásobníku L identických Transformer enkodér vrstev. Každý enkodér blok navíc obsahuje dvě subvrstvy. První z nich je multi-head self-attention mechanismus a druhá z nich je jednoduchá fully-connected dopředná síť.

Tvůrci BERT modelu představili ve svém článku [14] v roce 2018 dvě základní verze:

- BERT_{BASE}: 12 vrstev (enkodér bloků), 12 attention hlav, 110 milionů parametrů, 768 skrytých vrstev
- BERT_{LARGE}: 24 vrstev (enkodér bloků), 16 attention hlav, 340 milionů parametrů, 1024 skrytých vrstev

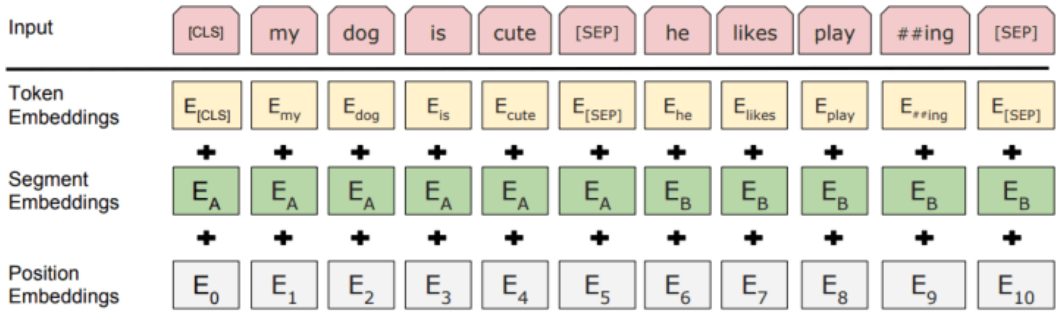
Jednotlivé vrstvy jsou zapojeny za sebou a tvoří dohromady celý BERT model.

2.5.1.2 Předzpracování textu

Vývojáři, kteří stáli za vznikem BERT modelu, určili i specifická pravidla, kterých je potřeba se držet při tvorbě vstupních vektorů. Vstupem BERT modelu je kombinace následujících vektorů:

- **poziční vektor**: využit pro určení pozice tokenu ve větě
- **vektor označující segmenty**: BERT model je taktéž schopný zpracovat dvě věty na vstupu jako pár. Vektor rozlišuje mezi větou A a větou B (viz. *segment embeddings* v obrázku 2.11). Hodí se například při úlohách Question-Answering.
- **vektor tokenů**: vektory obsahující specifické tokeny, které jsou definované tzv. WordPiece tokenizací - viz. níže

Ukázka zpracování vstupního textu a příslušných vektorů je na obrázku 2.11.



Obrázek 2.11: Vstupní vektory. [17]

Na předešlém obrázku je v prvním řádku (červené bloky) vidět rozdělené slovo *playing* na dva tokeny *play*, *##ing*. Tyto tokeny jsou získány již zmíněnou WordPiece Tokenizací. Zároveň je možné si všimnout pomocných tokenů. Token $[CLS]$ je použit vždy na začátku věty. Následují jednotlivé věty oddělené pomocí tokenu $[SEP]$. Jelikož BERT model očekává vektory fixní délky, je zbytek nevyužitých pozic vyplněn tokeny $[PAD]$. [18]

WordPiece model

WordPiece je algoritmus tokenizace používaný v úlohách zpracování přirozeného jazyka. Jedná se o interní kód firmy Google. Volně dostupnou variantou je tzv. SentencePiece tokenizace. [17] WordPiece je v podstatě model, jež vytvoří slovník pevné velikosti skládající se z jednotlivých znaků, podslov a slov, které nejlépe vystihují daný jazykový korpus. Trénovací algoritmus je následovný:

1. příprava velkého textového korpusu
2. určení požadované velkosti slovníku

3. slovník WordPiece modelu je inicializován základní sadou znaků - například je možné využít všechny znaky, které český jazyk využívá

4. rozdělení trénovacího korpusu na jednotlivé tokeny obsažené ve slovníku

5. natrénování jazykového modelu pro tokenizovaný korpus

6. spojování tokenů slovníku takovým způsobem, aby došlo k co největšímu zlepšení úspěšnosti modelu na trénovacích datech.

7. návrat ke kroku 3 - pokud není splněna podmínka úspěšnosti, nebo dokud není ve slovníku dostatek slov

Výhodou WordPiece tokenizace je pokrytí většiny možných slov, protože tokeny jsou získány trénováním na velkých trénovacích korpusech v daném jazyce. [19]

2.5.1.3 Předtrénování modelu

BERT model je předtrénován pomocí dvou různých úloh strojového zpracování přirozeného jazyka. První úlohou je predikce slova na základě slovního kontextu a návaznosti vět. Pro úlohu predikce slova se využívá tzv. proces maskování. Patnáct procent tokenů z trénovacího korpusu je změněno následujícím postupem:

- v 80% případů je token nahrazen maskovacím tokenem $[MASK]$
- v 10% případů je slovo nahrazeno náhodným slovem ze slovníku
- v 10% případů je slovo necháno nezměněné

Proces předtrénování se snaží odhadnout originální hodnotu změněného tokenu na základě kontextu ostatních slov ve větě. Druhou úlohou předtrénování je odhad návaznosti dvou vstupních vět. BERT model se snaží při trénování na této úloze určit, zdali dvě věty, které jsou na vstupu modelu, jsou ve správném pořadí, tzn. zdali se takovém pořadí vyskytují i v trénovacím korpusu. Během trénování je vybráno 50% dvojic vět, jež jsou ve správném pořadí, a 50% dvojic, které nejsou ve správném pořadí, resp. druhá věta je vybrána náhodně z trénovacího korpusu. Pro rozlišení dvou vět se používají speciální tokeny popsané v části předzpracování textu. [15]

Popsané předtrénování modelu je velice výpočetně náročné. Autoři myšlenky však s vydáním článku zveřejnili i několik předtrénovaných modelů, které lze využít. Předtrénované modely se liší v počtu enkodér bloků a také v počtu attention hlav. Dvě základní varianty byly zmíněny v části 2.5.1.1. Jedná se o modely BERT-Large a BERT-Base. Avšak existuje celá řada dalších modelů. Při použití programovacího jazyka Python se například nabízí využití knihovny Transformers od komunity Hugging Face [20]. Jedná se knihovnu poskytující přes 30 různých předtrénovaných transformer modelů (BERT, ALBERT, RoBERTa, atd.). Modely lze využít i pro úlohy strojového zpracování českého jazyka. Příkladem je použití modelu *bert-base-multilingual-cased*, který byl využit i v rámci této diplomové práce.

2.5.1.4 Fine-Tuning

Fine-tuning je tzv. vylepšení modelu. Předtrénované BERT modely je možné využít na celou řadu specifických úloh. K BERT modelu je připojena vrstva podle dané úlohy. Například pro úlohy klasifikace se jedná o jednoduchou klasifikační vrstvu. Při fine-tuning trénování většina hyperparametrů BERT modelů zůstává stejná a model se pouze přizpůsobuje účelům dané úlohy. Autorům BERT modelu se podařilo dosáhnout state-of-the-art výsledků v široké paletě úloh zpracování přirozeného jazyka právě zmíněným fine-tuning trénováním. [15]

Fine-tuning předtrénovaných modelů pro různé úlohy strojového zpracování přirozeného jazyka byl náplní této diplomové práce. Těmto úlohám budou věnovány závěrečné kapitoly.

2.6 Model T5

Jedním z nejnovějších modelů založených na transfer learningu[3] je tzv. **T**ext-**T**o-**T**ext **T**ransfer **T**ransformer model, zkráceně T5. Tento model byl představen v článku *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*[21] na sklonku roku 2019.

Zmíněný článek je především průzkumem použití moderních přístupů pro zpracování přirozeného jazyka. Pro testování odlišných přístupů byly natrénovány modely na rozdílných datasetech a s různými trénovacími strategiemi. Autoři navíc navrhli framework, který se pokouší sjednotit všechny „jazykové" úlohy do podoby transformace textu na text. [22]

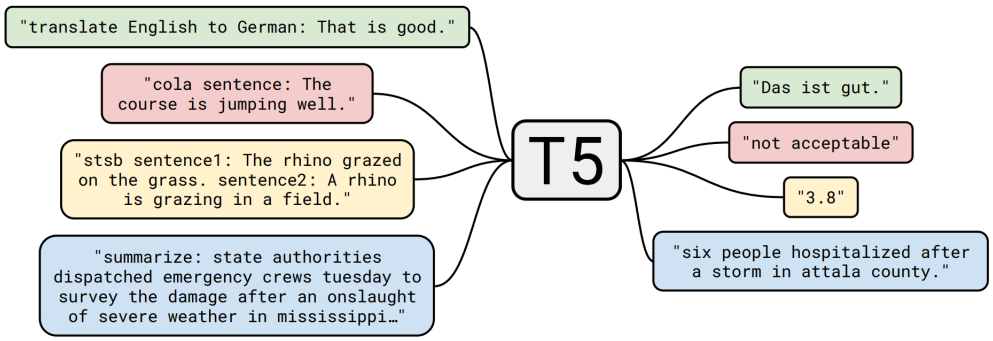
2.6.1 Text-to-Text Framework

Modely s architekturou podobnou modelu BERT jsou předtrénovány na úlohách, jež byly zmíněny v sekci 2.5.1.3, tj. predikce slova na základě slovního kontextu a návaznost vět. Následně jsou tyto předtrénované modely dotrénovány na specifické úlohy - například pro predikci třídního označení při úloze klasifikace textu.

Text-to-Text framework naopak navrhuje použít u všech úloh zpracování přirozeného jazyka stejný model, stejnou ztrátovou funkci a stejné hyperparametry. Pro tento přístup jsou vstupní data v takové formě, aby byl model schopný rozeznat, o jakou úlohu se jedná. Výstupem je pak jednoduše textová forma očekávaného výstupu. Princip T5 modelu je graficky znázorněn na obrázku 2.12. Jak je z obrázku patrné, aby bylo možné využít stejný model pro více specifických úloh, je zapotřebí

[3]Informace získaná při řešení jednoho problému je použita pro řešení problému druhého - například využití informace získané při trénování rozpoznávání osobních aut může být použita pro rozpoznávání nákladních aut.

přidat textový prefix, který určuje typ úlohy. Tento prefix je zahrnut mezi hyperparametry modelu. [22]

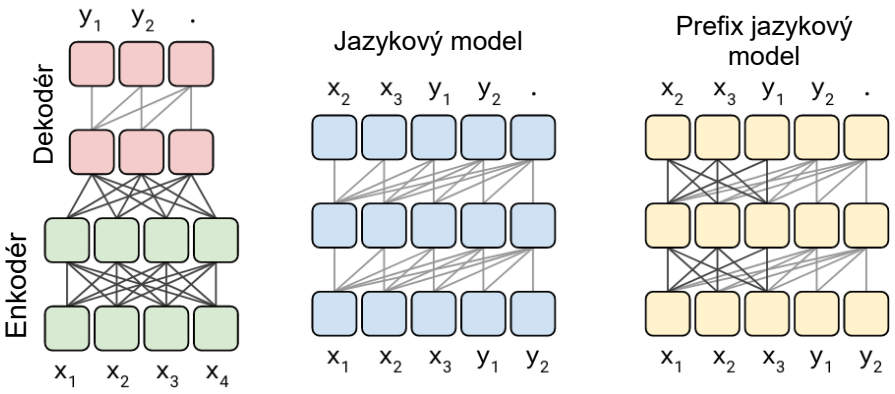


Obrázek 2.12: Princip T5 modelu. [23]

2.6.2 Architektura modelu

Autoři článku experimentovali s různými variantami architektur modelů. Pro testování byly uvažovány následující přístupy:

- **enkodér-dekodér**: klasická seq2seq[4] architektura, kde enkodér je trénován tzv. fully-visible způsobem[5] a dekodér je trénován tzv. kauzálním způsobem[6]
- **jazykový model**: v podstatě kauzální způsob jako u enkodér-dekodér přístupu (jedná se o přístup autoregresního modelování)
- **prefix jazykový model**: kombinace obou předešlých přístupů



Obrázek 2.13: Jednotlivé architektury modelů. [21]

[4]Jedná se o machine learning přístup zpracování dat
[5]Každý token přispívá k výpočtu attention všech ostatních tokenů v sekvenci (viz.2.3.2).
[6]Každý token může přistupovat k tokenům, které se vyskytují v sekvenci dříve.

I navzdory faktu, že enkodér-dekodér model používá dvakrát více parametrů než pouhý enkodér (BERT), nebo dekodér model (jazykový model), mají všechny modely podobnou výpočetní náročnost. Oba bloky modelu, tj. enkodér a dekodér jsou svou velikostí a konfigurací podobné modelu BERT_{BASE}. V článku je zároveň dokázáno, že sdílení parametrů enkodéru a dekodéru nevede při půlení celkového počtu parametrů k podstatnému úpadku výkonu modelu. Enkodér-dekodér architektura dosáhla nejlepších výsledků pro text-to-text framework. [21] [22]

2.6.3 Předtrénování modelu

Autoři článku se taktéž zaměřili na různé přístupy unsupervised learningu[7]. Pro předtrénování použitých modelů využili hned několik možných variant trénování. Několik z nich je znázorněno v tabulce 2.1.

Přístup	Vstup	Očekávaný výstup
Prefix LM[8]	Děkuji ti za pozvání	na tvoji oslavu minulý týden.
BERT-style [14]	Děkuji ti <M><M> na tvoji oslavu jablko týden.	originální text
Přeskupení	oslavu minulý Děkuji na tvoji ti pozvání za týden.	originální text
MASS-style [24]	Děkuji ti <M><M> na tvoji oslavu <M> týden.	originální text
Nahrazení mezer	Děkuji ti <X>na tvoji oslavu <Y>týden.	<X>za pozvání <Y>minulý <Z>
Vynechání tokenů	Děkuji ti na tvoji oslavu týden.	za pozvání minulý
Náhodné mezery	Děkuji ti <X> tvoji <Y> týden	<X> za pozvání na <Y> oslavu minulý <Z>

Tabulka 2.1: Ukázka použitých přístupů při předtrénování modelu pomocí učení bez učitele. Jednotlivé přístupy jsou aplikovány na větu *Děkuji ti za pozvání na tvoji oslavu minulý týden.* <M> jsou tokeny reprezentující masku. [21]

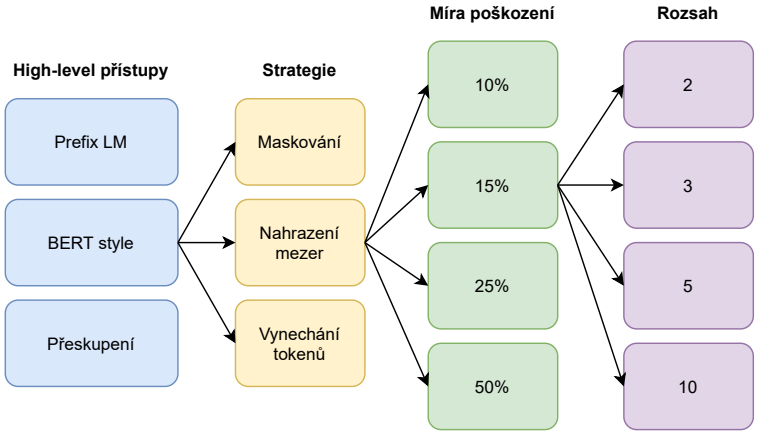
Z jednotlivých testování vycházely nejlépe experimenty založené na BERT-style [14] přístupu viz. tabulka 2.1. Dále byl zkoumán vliv podílu "poškozených"[9] vstupních tokenů a těch neupravených. Nejlepších výsledků bylo dosaženo při použití 15% "poškozených" vstupních tokenů. Následně byla měněna délka porušených částí, tj. kolik tokenů vyskytujících se za sebou je poškozeno. Příkladem je vstupní sekvence obsahující 500 tokenů. Pokud je zvolen *corruption rate*[10] 15% a počet mezer pro sekvenci rovno 25, pak bude celkově poškozeno $500 \cdot 0,15 = 75$ tokenů a průměrný rozsah, tj. počet po sobě jdoucích poškozených tokenů, bude $75/25 = 3$. Různé varianty nastavení těchto parametrů vedou na rozdílné délky trénovacích sekvencí. Ruku v ruce se s tím také mění rychlost trénování. V článku doporučují měnit parametry BERT-style přístupu trénování podle dostupných výpočetních zdrojů. Na obrázku je 2.14 je vývojový diagram popisující postup při trénování T5 modelu popsaný v této části. [21] [22]

[7] učení bez učitele

[8] language modelling - česky jazykové modelování

[9] Poškozené tokeny jsou ty, které jsou například nahrazeny maskovacími tokeny <M>.

[10] míra poškozených tokenů v sekvenci



Obrázek 2.14: Vývojový diagram trénování T5 modelu. Použito bylo několik různých přístupů. [21]

2.6.4 Textový korpus pro předtrénování modelu

Stejně jako technika předtrénování modelu, tak i dataset sloužící k předtrénování modelu je důležitou součástí učení založeném na principu transfer learningu. Stereotypem v předtrénování takovýchto modelů je využití obsáhlých neoznačených trénovacích korpusů, tj. korpusů bez informace od učitele. Mezi takové datasety patří i *Common Crawl corpus*[11] obsahující petabyty textů ze všech různých webových stránek za posledních 12 let, tj. přibližně 20TB dat každý měsíc.[25] Data se získávají přímo z HTML souborů. Avšak takový dataset obsahuje velké množství přebytečných textů - jako menu jednotlivých webových stránek, duplikované texty, chybové hlášky a další. Tvůrci článku [21] tento korpus pročistili v několika směrech a vytvořili korpus C4 (**C**olossal **C**lean **C**rawled **C**orpus) [22]:

1. Uvažování pouze vět, které jsou ukončeny interpunkčním znaménkem (tečka, vykřičník, otazník nebo koncová uvozovka).

2. Odstranění všech stránek, které obsahují nějaké urážlivé slovo. Jako reference byl použit soupis nevhodných slov z GitHub repozitáře *List of Dirty, Naughty, Obscene and Otherwise Bad Words* [26]. Tento repozitář obsahuje seznamy hanlivých slov ve 27 jazycích.

3. Odstranění všech chybových hlášek typu *JavaScript must be enabled*, tj. odstraněny jsou všechny řádky obsahující slovo JavaScript.

4. Odstranění stránek jejichž *placeholder* atribut obsahuje text jako "*lorem ipsum*".

5. Odstranění stránek, které obsahují složené závorky "{" (jelikož většina dnešních programovacích jazyků obsahuje právě složené závorky).

[11] `https://commoncrawl.org/the-data/`

6. Odstranění duplikovaných třech po sobě jdoucích vět, tzn. pokud se objeví tři věty jdoucí za sebou několikrát ve stejném pořadí, pak se duplikáty odstraní.

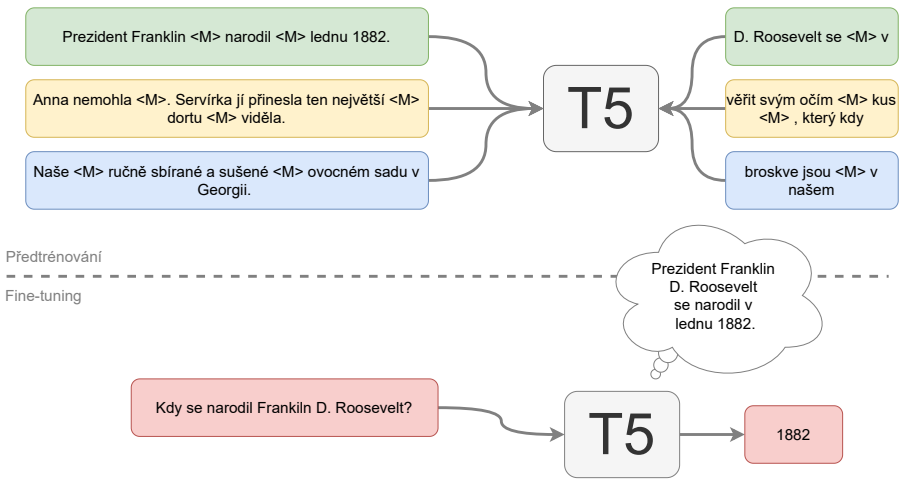
7. Vyfiltrování ne-anglických stránek.

Tento postup filtrování *Common Crawl* korpusu vedl k vytvoření již zmíněného 750GB velkého C4 korpusu, který je svou velikostí větší než většina trénovacích datasetů určených pro předtrénování transfer learning modelů a navíc obsahuje relativně "čistý" text. Tento textový korpus byl použit pro předtrénování modelu $T5_{BASE}$, který byl použit v této práci. [22]

2.6.5 Použití modelu

Autoři článku [14] provedli několik dalších experimentů pro zjištění nejlepších dosavadních přístupů, jež vedou k získání state-of-the-art výsledků pro co nejvíce úloh zpracování přirozeného jazyka. Například provedli experimenty s různými trénovacími strategiemi, s rozdílnými velikostmi modelů a s použitím odlišných počtů epoch. Poznatky získané ze všech zmíněných experimentů použili pro předtrénování T5 modelu.

Framework T5 je možné použít pro nespočet specifických úloh. Jednou z úloh může být odpovídání na otázky (question-answering). Princip této úlohy s využitím T5 modelu je zobrazen na obrázku 2.15. Tento model je novinkou v přístupech ke zpracování přirozeného jazyka a byl vybrán pro srovnání výkonnosti s modelem BERT. [21] [22]



Obrázek 2.15: V průběhu předtrénování se T5 model naučí zaplnit mezery v textu (označeny <M>) z dokumentů v korpusu C4. Následně je možné pomocí fine-tuning použít T5 model pro question-answering úlohu. Otázky jsou však pouze v omezeném rozsahu. [22]

3 Vývojové prostředí

Následující část je věnována frameworkům použitým při jednotlivých experimentech. Popsány jsou taktéž parametry použitého výpočetního prostředí.

3.1 Framework

Trénování hlubokých neuronových sítí je výpočetně náročné a velice složité na implementaci. V dnešní době však lze využít hned několika frameworků usnadňujících práci s neuronovými sítěmi. Použití frameworků je možné i bez hlubší znalosti jednotlivých algoritmů. Pro fine-tuning byly použity frameworky TensorFlow a PyTorch.

3.1.1 TensorFlow

TensorFlow byl vyvinut společností Google a veřejně vydán v roce 2015.[27] TensorFlow je kompatibilní s několika programovacími jazyky, avšak nejpoužívanější variantou je kombinace s programovacím jazykem Python. Zároveň tento framework umožňuje pouze s minimální změnou spustit trénovací algoritmy na CPU i GPU, [1] a dokonce i na odlišných platformách (mobilní telefon, počítač, atd.). [28]

3.1.1.1 Principy

Jedná se o open-source platformu, která v první funkční verzi využívala tzv. toku dat v orientovaných grafech. Z tohoto principu plyne i název *TensorFlow*, tj. tok tenzorů. Tenzory jsou multidimenzionální vektory obsahující potřebná data k výpočtům. TensorFlow verze 1.x sestávaly ze dvou bloků - knihovny definující tzv. výpočetní grafy a z modulu pro zpracování takových grafů na různých hardwarech.

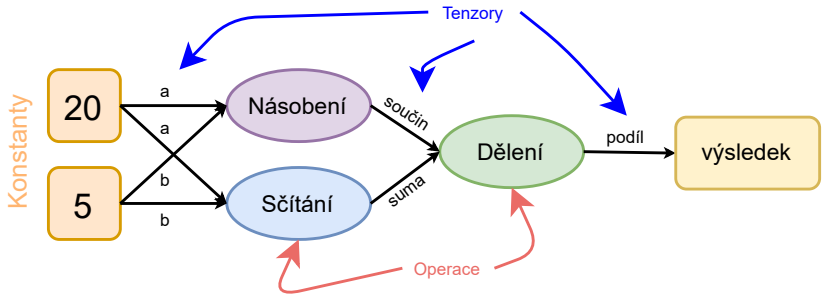
Výpočetním grafem se rozumí orientovaný graf, jehož uzly tvoří jednotlivé operace, proměnné a zástupné symboly. Operacemi je myšleno vytváření dat a manipulace s nimi. Proměnné na druhé straně reprezentují sdílený stav, se kterým je možné manipulovat právě pomocí jasně daných operací. Hranám grafu odpovídají data neboli tenzory, jež "proplouvají" přes jednotlivé operace.

[1] centrální a grafická procesorová jednotka

Takový graf je jednoduché znázornit graficky. Příklad je prezentován na obrázku 3.1. Tento graf reprezentuje algoritmus vytvoření dvou konstant. Součin a suma těchto konstant jsou vyděleny mezi sebou. Tím je získána výsledná hodnota. Jedná se pouze o jednoduchou ilustraci. Sčítání a násobení obou konstant je nezávislé a je možné je dělat ve stejný čas. TensorFlow verze 1.x byl díky výpočetním grafům schopný rozdělit takto nezávislé úlohy mezi více GPU, nebo je dokonce rozdělit na více výpočetních strojů. Celý proces tedy spočívá ve vytvoření grafu, který je následně modulem pro zpracování grafů rozplánován, a dílčí úlohy jsou vykonány. Modul zároveň alokuje paměť pro jednotlivé výpočty. Ukázka zdrojového kódu tvorby grafu a spuštění modulu pro zpracování vytvořeného grafu ve frameworku TensorFlow verze 1.x je vidět v přiloženém kódu 3.1.

Další klíčovou výhodou je přenositelnost. Výpočetní graf je nezávislý na programovacím jazyku. Je tedy možné graf vytvořit pomocí jayzku Python, uložit model a obnovit ho v odlišném jazyce, například C++, pokud je zapotřebí vysoká rychlost výpočtů. [28][29]

Verze TensorFlow 2 umožnila přechod k tzv. eager execution výpočetnímu prostředí, které hodnotí operace okamžitě. Stále je však možné využít i výpočetních grafů.



Obrázek 3.1: Výpočetní graf jednoduchých matematických operací.

```python
import tensorflow as tf

with tf.Session() as sess:
    # Faze 1: tvorba vypocetniho grafu
    a = tf.constant(20, name="a")
    b = tf.constant(5, name="b")
    prod = tf.multiply(a, b, name="Multiply")
    _sum = tf.add(a, b, name="Add")
    res = tf.divide(prod, _sum, name="Divide")

    # Faze 2: Spusteni modulu pro zpracovani grafu
    out = sess.run(res)
```

Kód 3.1: Ukázka zdrojového kódu v programovacím jazyce Python

3.1.2 PyTorch

Framework PyTorch byl vyvinut společností Facebook v roce 2016. Byl vytvořen za účelem poskytnout optimalizaci podobně jako TensorFlow, avšak s jednodušší tvorbou modelů. PyTorch je však zatím rozšířen především oblasti výzkumu nežli v produkci jako je tomu u TensorFlow. [29]

3.1.2.1 Principy

Jak název napovídá, PyTorch je postaven na výpočetním frameworku Torch, který poskytuje rychlé výpočetní algoritmy psané v programovacím jazyce C. PyTorch se chová jako tzv. wrapper[2] a poskytuje Python rozhraní přistupovat k algoritmům psaným právě v programovacím jazyce C. Díky tomu, že je vše psané v nízkoúrovňovém jazyce C, je možné tvořit vysoce propracované struktury neuronových sítí v Pythonu bez použití nízkoúrovňových funkcí. TensorFlow 2.0 se nechal inspirovat právě frameworkem PyTorch a v dnešní době jsou tyto dva frameworky velice podobné. [29]

3.2 Výpočetní prostředí

Pro trénování byly využity výpočetní zdroje organizace MetaCentrum VO[3]. Především pak Konos cluster výzkumného centra NTIS a katedry kybernetiky FAV ZČU. Každý fyzický stroj tohoto clusteru má k dispozici 20 CPU a 4 GPU.

Při experimentech byl použit programovací jazyk Python. Pro tvorbu zdrojového kódu byla využita open-source webová aplikace Jupyter Notebook[4]. Jedná se o nástroj, který uživatelům poskytuje interaktivní prostředí s použitím webového rozhraní.

[2] Návrhový vzor umožňující spolupráci tříd, které by kvůli odlišným rozhraním spolupracovat nemohly. [30]

[3] `https://metavo.metacentrum.cz/cs/`

[4] `https://jupyter.org/`

4 Experimenty

Pro srovnání BERT modelu a modelu T5 bylo provedeno několik experimentů. Oba modely byly testovány na dvou úlohách zpracování přirozeného jazyka, a to analýze sentimentu a detekci pojmenovaných i sémantických entit. Pro experimenty byly využity volně dostupné trénovací datasety pro anglický a český jazyk.

4.1 Analýza sentimentu

Analýza sentimentu jako taková spadá pod úlohy zpracování přirozeného jazyka. Význam slova sentiment je "silné a bezprostřední hnutí mysli" nebo také cit[1]. Pro tuto úlohu se však jedná především o identifikaci a extrakci vztahu autora k předmětu textu. Názor je subjektivní informací, která je zanesena v textu. Samotný sentiment lze rozdělit do několika tříd. Buď se používá dělení do pěti tříd - *vysoce negativní, negativní, neutrální, pozitivní a vysoce pozitivní*, anebo lze krajní třídy zanedbat a určovat pouze třídy *negativní, neutrální a pozitivní*. Toto rozdělení většinou závisí na dostupných třídách v trénovacích datech. Pro účely této diplomové práce byly uvažovány dokonce pouze třídy dvě - *pozitivní a negativní*. Úloha analýzy sentimentu se používá například v sociálních médiích. Pomocí modelů pro analýzu sentimentu lze kupříkladu určit sentiment příspěvku nebo komentáře přidaného na sociální síti Facebook. [31]

4.1.1 Dataset IMDB recenzí

Pro úlohu analýzy sentimentu byl nejprve využit anglický dataset obsahující recenze filmů [32] z internetové databáze IMDB.

4.1.1.1 Trénovací data

Dataset IMDB je určený pro binární klasifikaci sentimentu, tj. rozeznávání pozitivní a negativní recenze. Tvůrci tohoto datasetu poskytují 25 000 vysoce polarizovaných[2] recenzí pro trénování a 25 000 recenzí pro testování. Jednotlivé recenze jsou

[1] Citováno ze stránky: `https://lidovyslovnik.cz/index.php?dotaz=sentiment`.

[2] Lze přesně určit o jakou třídu sentimentu se jedná.

rozděleny do samostatných souborů. Tyto soubory jsou dále rozčleněny podle sentimentu do složek negativních a pozitivních recenzí. Trénovací data byla rozdělena v poměru 1:5 mezi trénovací a validační data. Průměrná délka recenze ve smyslu počtu slov je pro trénovací sadu 233,8 slov a pro testovací sadu 228,5 slov.

Ukázka pozitivní recenze:
Very good drama although it appeared to have a few blank areas leaving the viewers to fill in the action for themselves. I can imagine life being this way for someone who can neither read nor write. This film simply smacked of the real world: the wife who is suddenly the sole supporter, the live-in relatives and their quarrels, the troubled child who gets knocked up and then, typically, drops out of school, a jackass husband who takes the nest egg and buys beer with it. 2 thumbs up.

	Pozitivní	Negativní
Trénovací	12500	12500
Testovací	12500	12500

Tabulka 4.1: Počet dostupných trénovacích a testovacích dat pro IMDB dataset.

4.1.1.2 BERT model

Pro účely této úlohy byl použit předtrénovaný model poskytnutý tvůrci BERT modelu [14]. Jedná se o variantu BERT-Base[3]. BERT je schopný pracovat se sekvencemi o velikosti až 512 tokenů. Z důvodu nedostatku výpočetní paměti na grafických kartách však byla nejprve omezena délka vstupní sekvence na 128 tokenů. Některé recenze jsou však delší a vstupní data bylo potřeba nejprve omezit, tj. vzít pouze prvních 128 tokenů. Samotná posloupnost tokenů byla získána WordPiece tokenizátorem, zmíněným v části 2.5.1.2. Model byl optimalizován na úlohu binární klasifikace, tzn. na výstup předtrénovaného modelu byla přidána jednoduchá fully-connected vrstva s aktivační funkcí softmax. Výstupem je dvoudimenzionální vektor. Pomocí matematické operace *argmax* je získána predikce třídy recenze.

Takto připravený model byl testován, avšak v průběhu testování se ukázalo, že omezení délky vstupní sekvence vede k výrazně horším výsledkům. Proto byly využity výpočetní zdroje online vědecké komunity Kaggle[4]. Pro registrované uživatele je zde k dispozici výpočetní prostředí, které umožňuje využití tensorových procesorových jednotek (TPU) pro machine learning úlohy. Díky TPU jednotkám bylo možné využít celý vstupní rozsah BERT modelu, tj. 512 tokenů.

Tvůrci BERT modelu zároveň zveřejnili hodnoty hyperparametrů, se kterými dosáhli state-of-the-art výsledků. Tyto hodnoty byly použity jako výchozí. Výsledky jsou graficky i numericky znázorněny na konci této části.

[3]označení modelu: Uncased_L-12_H-768_A-12 [33]
[4]`https://www.kaggle.com/`

4.1.1.3 T5 model

Druhým použitým modelem pro tuto úlohu byl T5 model popsaný v části 2.6. Přesněji řečeno byl použit $T5_{BASE}$ model předtrénovaný na anglickém C4 korpusu [21]. T5 architektura umožňuje generovat jakýkoliv textový výstup, tzn. výstupem této architektury byly slovní označení *positive* nebo *negative*. Vstupní text nebylo potřeba nijak omezovat. Pro snadnější testování T5 modelu byla využita knihovna T5s implementována panem doktorem Janem Švecem [34]. Tato knihovna umožňuje pomocí jednoduchého konfiguračního souboru měnit parametry modelu, zároveň obsluhuje celý proces trénování.

4.1.2 Dataset ČSFD recenzí

Druhým experimentem bylo testování modelů BERT a T5 pro potřeby analýzy sentimentu na datech obsahující recenze z česko-slovenské filmové databáze[5].

4.1.2.1 Trénovací data

ČSFD dataset je volně dostupný korpus poskytnutý spolu s článkem *Sentiment Analysis in Czech Social Media Using Supervised Machine* [35]. Obsahuje přibližně 90 tisíc recenzí v českém jazyce. Tyto recenze jsou rozděleny do tří tříd - negativní, neutrální a pozitivní. Pro experimenty byly však opět uvažovány pouze dvě třídy - pozitivní a negativní. Trénovací data byla rozdělena v poměru 1:5 na data trénovací a validační. Průměrná délka recenze ve smyslu počtu slov je pro trénovací sadu 49,4 slov a pro testovací sadu 51,6 slov.

Ukázka pozitivní recenze:
Pro mě osobně velké překvapení. Před shlédnutím filmu jsem o Oliveru Twistovi nevěděl nic.

	Pozitivní	Negativní
Trénovací	30397	29216
Testovací	500	500

Tabulka 4.2: Počet dostupných trénovacích a testovacích dat pro ČSFD dataset.

4.1.2.2 BERT model

Úloha analýzy sentimentu na datasetu tvořeného z českých filmových recenzí byla testována hned se dvěma různými modely typu BERT. Prvním testovaným modelem byl volně dostupný multi-jazykový model $BERT_{BASE}$ Multilingual Cased [14]. Tento model je předtrénován na korpusu složeném z článků online encyklopedie Wikipedia[6]. Bylo použito 104 největších různojazyčných Wikipedií. BERT

[5] webové stránka ČSFD: `https://www.csfd.cz/`
[6] `https://cs.wikipedia.org`

multilingual model je možné použít jen s lehkými úpravami pro širokou paletu úloh zpracování přirozeného jazyka, a to ve všech 104 jazycích. Trénování probíhá stejně jako u klasické BERT architektury (viz. část 2.5.1.3), tj. s využitím úlohy predikce slova na základě slovního kontextu, s využitím maskování a úlohy návaznosti vět. Pro tokenizaci vstupní sekvence byl využit WordPiece tokenizátor. [14]

Tento model by obdobně jako model pro analýzu sentimentu IMDB recenzí optimalizován na úlohu binární klasifikace s omezením vstupu na 128 tokenů, resp. 512 tokenů při použití TPU jednotek.

Druhým modelem byl BERT model předtrénovaný pouze na českém trénovacím korpusu složeném z textů webových stránek, viz. článek *Czech BERT Models for Multi-label Document Classification* [36]. Architektura modelu je však shodná s architekturou modelu BERT_{BASE}. Jediným rozdílem je použití SentencePiece tokenizátor oproti WordPiece tokenizátoru, který byl použit pro vícejazyčný BERT model. Jelikož "český" a Multilingual BERT model vychází ze stejné architektury, je možné porovnat výsledky získané těmito modely s různými nastaveními hyperparametrů. Výsledky těchto experimentů budou znázorněny na konci této části.

4.1.2.3 T5 model

Pro analýzu sentimentu českého jazyka byl použit T5 model s architekturou shodnou s modelem popsaným v části 2.6. Liší se však korpusy, na kterých byly tyto modely předtrénovány. Model použitý pro tento experiment byl předtrénován pouze na české části *Common Crawl* korpusu [25], tzn. byla použita trénovací data obsahující pouze české texty. Následující filtrování textů bylo shodné s postupem v části 2.6.4.

4.1.3 Výsledky

BERT modely byly testovány s mnoha kombinacemi hodnot hyperparametrů. Doporučené hodnoty (zvýrazněny tučně) podle článku *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding* [14] byly doplněny o několik dalších. Celkem se jedná o 90 kombinací:

- počet epoch: 2, **3**, **4**
- learning rate[7]: $1 \cdot 10^{-5}$, $\mathbf{2 \cdot 10^{-5}}$, $\mathbf{3 \cdot 10^{-5}}$, $\mathbf{4 \cdot 10^{-5}}$, $\mathbf{5 \cdot 10^{-5}}$
- batch size[8]: **16**, **32**, 64
- délka sekvence: 128, 512

[7] Jedná se o parametr určující velikost kroku při hledání minima ztrátové funkce.

[8] Tento hyperparametr určuje pro kolik dat najednou se počítá gradient, podle něhož dochází k úpravě vah.

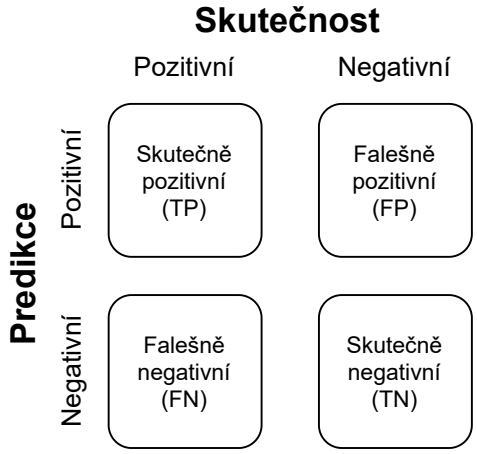
Pro T5 model nejsou žádné doporučené hodnoty hyperparametrů a bylo potřeba je nalézt experimentálně.

Nastavení hyperparametrů T5 modelu však nelze přímo porovnat s hyperparametry BERT modelu. Pro T5 model byla použita již zmíněná knihovna *t5s*. Tato knihovna využívá pro fine-tuning T5 modelů jednoduchý konfigurační soubor. V tomto souboru se nastavuje počet epoch, learning rate a další hyperparametry modelu. Místo parametru *batch_size* se však používá parametr *steps_per_epoch*. Tento parametr určuje, kolik vstupních sekvencí je zpracováno v průběhu jedné epochy, navíc je však brána v potaz i proměnná délka sekvencí. Batch_size v tomto případě není konstantní a je úměrný délkám sekvencí.

4.1.3.1 Evaluační metriky

Pro zhodnocení výsledků jednotlivých modelů je potřeba zavést některé metriky, které určí výkonnost daného modelu v porovnání s ostatními. K úloze analýzy sentimentu bylo přistupováno jako k jednoduché binární klasifikaci, tj. rozdělení na pozitivní a negativní recenze, ať už v českém nebo anglickém jazyce. Pro binární klasifikaci mohou nastat 4 stavy výsledku. Pokud model správně zařadí recenzi do jedné ze tříd (pozitivní, negativní), pak se jedná o tzv. pozitivní rozpoznání (TP-pravdivě pozitivní, TN-pravdivě negativní). Pokud však dojde k nesprávné klasifikaci, tzn. pozitivní recenze je označena jako negativní, resp. negativní recenze je označena jako pozitivní, pak dochází k chybnému rozpoznání (FP-falešně negativní, FN-falešně pozitivní).

První možnou metrikou pro vyhodnocení úspěšnosti modelu je tzv. chybová matice (též konfusní matice). Jedná se o matici obsahující ve sloupcích skutečný sentiment recenze a řádky obsahují hodnotu odhadovaného sentimentu, viz. obrázek 4.1.



Obrázek 4.1: Chybová matice pro vyhodnocení úlohy binární klasifikace.

Další dvě metriky se sebou úzce souvisí. Jedná se o přesnost a úplnost. První zmíněná metrika je dána podílem počtu skutečně pozitivních recenzí a počtem všech

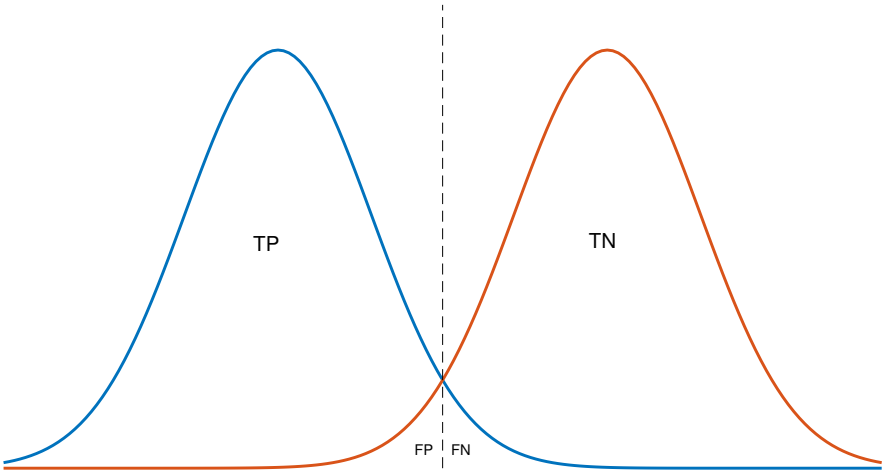
pozitivně označených recenzí, tj. i těch, které jsou nesprávně označeny jako pozitivní. Jinými slovy lze tato metrika definovat jako pravděpodobnost, se kterou jsou pozitivně označené recenze opravdu pozitivní.

$$\text{Přesnost} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4.1}$$

Druhá zmíněná metrika je dána podílem počtu skutečně pozitivních recenzí a všech pozitivních recenzí, tj. i těch, jež byly označeny jako negativní. Tato metrika jinými slovy popisuje jak velká část pozitivních recenzí byla správně klasifikována.

$$\text{Úplnost} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.2}$$

Schopnost získat vysoké hodnoty přesnosti a úplnosti je vždy žádoucí, avšak v mnoha reálných situacích je zcela nemožné takového stavu dosáhnout. To vyplývá i z rovnic 4.1, 4.2 a z obrázku 4.2. Vždy záleží na typu úlohy, která je řešena. Někdy je potřeba zvýšit přesnost a někdy úplnost.



Obrázek 4.2: Na změnu přesnosti a úplnosti má vliv volba prahu, tj. šedá tečkovaná čára. Při posunu prahu doprava se zvýší úplnost a sníží přesnost, při posunu doleva je tomu naopak.

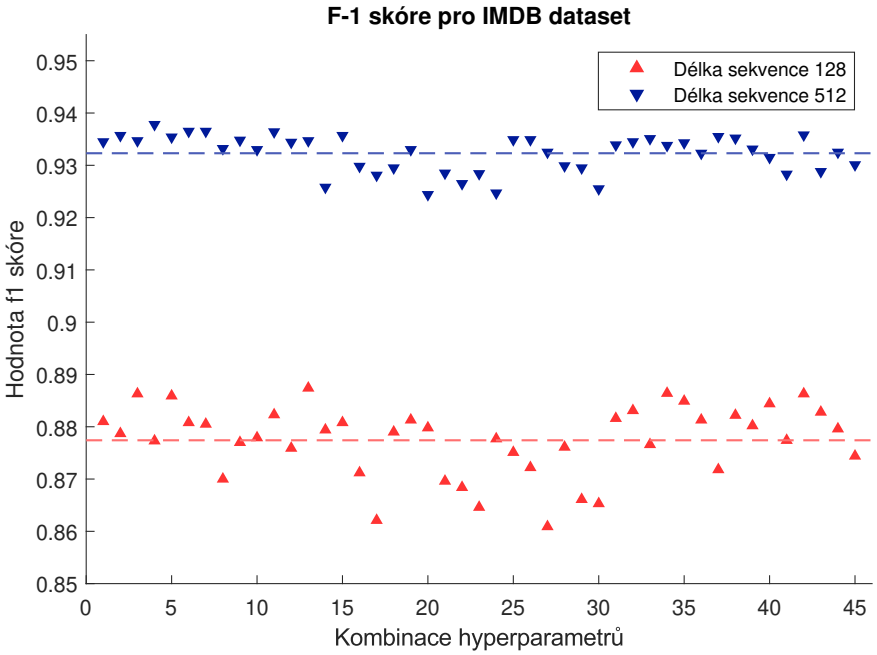
Poslední mírou, která vychází ze 4 stavů TP, TN, FP, FN je F-míra. Jedná se o harmonický průměr mezi přesností a úplností. Tato metrika udává celkový pohled na úspěšnost modelu.

$$\text{F-míra} = 2 \cdot \frac{\text{Přesnost} \cdot \text{Úplnost}}{\text{Přesnost} + \text{Úplnost}} \tag{4.3}$$

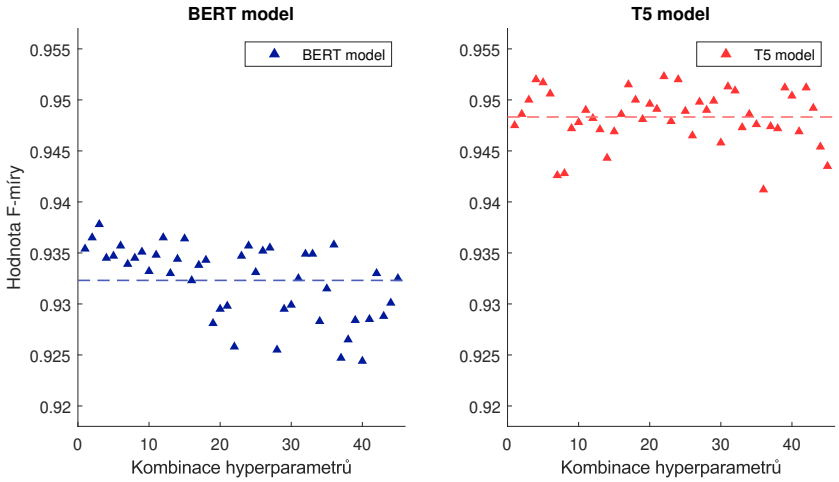
4.1.3.2 Analýza sentimentu IMDB recenzí

Pro srovnání výkonnosti BERT a T5 modelu na datasetu IMDB recenzí byly použity všechny zmíněné metriky v části 4.1.3.1. Nejprve došlo ke srovnání BERT modelů využívající vstupní sekvence velikost 128, resp. 512 tokenů. Z grafu 4.3 je patrné, že změna velikosti vstupní sekvence má pro tuto úlohu razantní vliv na výslednou hodnotu F-míry. To je dáno především tím, že většina recenzí z textového korpusu je delší než 128 tokenů (průměrná délka recenze v trénovacích datech je 233 slov), a tudíž dochází ke ztrátě informace cenné pro určení sentimentu celé recenze. Z grafu také vyplývá, že při použití délky sekvence 512 tokenů je menší variance v hodnotách F-míry, tj. menší rozptyl hodnot od střední hodnoty označené čárkovanou čárou.

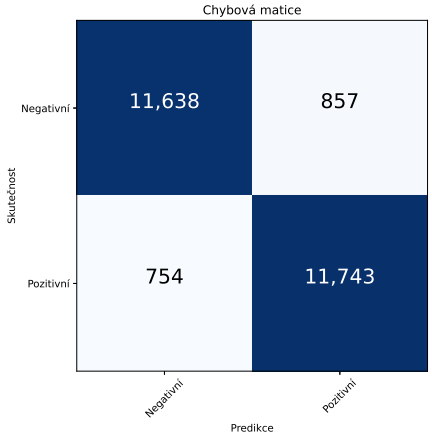
Na obrázku 4.4 jsou graficky zobrazeny všechny výsledné hodnoty F-míry modelů T5 a BERT (verze s délkou vstupní sekvence 512). Je patrné, že T5 model dosáhl výrazně lepších výsledků ohledně hodnoty F-míry. Takové tvrzení je zároveň číselně podloženo hodnotami v tabulce 4.3.



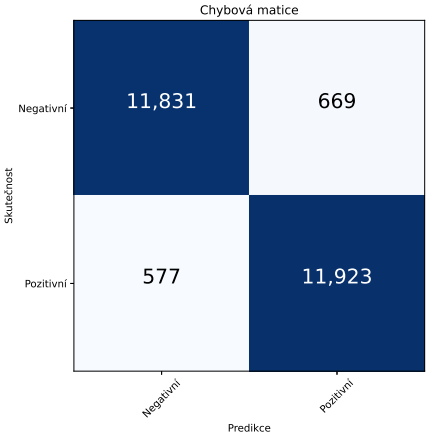
Obrázek 4.3: Vliv délky vstupní sekvence na výslednou hodnotou F-míry. Přerušovaná čára odpovídá střední hodnotě F-míry pro daný model.



Obrázek 4.4: Srovnání výsledků BERT a T5 modelu pro IMDB dataset. Přerušovaná čára odpovídá střední hodnotě F-míry pro daný model.



BERT model s hyperparametry:
počet epoch: 3, batch size: 32,
LR:$2 \cdot 10^{-5}$.



T5 model s hyperparametry:
počet epoch: 4, steps_per_epoch: 1400,
LR: $1 \cdot 10^{-4}$.

Obrázek 4.5: Srovnání chybových matic BERT a T5 modelu. U BERT modelu není součet všech pozitivních, resp. negativních recenzí roven hodnotě 12500, jak je uvedeno v tabulce 4.1, jelikož počet dat byl upraven tak, aby vyhovoval vztahu $počet\_dat \% batch\_size = 0$. Kde % je matematická operace modulo.

Na obrázku 4.5 jsou chybové matice BERT a T5 modelu pro specifické kombinace hyperparametrů. Je patrné, že jednotlivé sentimenty jsou chybně klasifikovány přibližně stejně, tzn. pozitivní recenze jsou označeny jako negativní v podobném počtu případů jako jsou negativní recenze označeny za pozitivní[9]. To jinými slovy

[9]V počtu 12500 pozitivních a 12500 negativních recenzí je rozdíl počtů chybně klasifikovaných zanedbatelný (rozdíl je přibližně 100 u obou modelů), průměrné hodnoty přesnosti a úplnosti v tabulce 4.3 toto tvrzení potvrzují.

znamená, že nedochází k chybné klasifikaci pouze jednoho sentimentu.

K numerickému srovnání BERT a T5 modelu bylo použito 30 nejlepších dosažených výsledků ve smyslu nejvyšší hodnoty F-míry. Srovnání obou modelů je v tabulce 4.3. Hodnoty potvrzují všechna předešlá tvrzení, a to především to, že T5 model dosáhl v této úloze výrazně lepších výsledků než model BERT.

BERT model (délka sek. 512)				T5 model			
Kombinace hyperparametrů	F-míra	Přesnost	Úplnost	Kombinace hyperparametrů	F-míra	Přesnost	Úplnost
počet epoch=2, LR=$1 \cdot 10^{-5}$, batch size=16	0.9378	0.9258	0.9519	počet epoch=4, LR=$1 \cdot 10^{-4}$, steps_per_epoch=1600	**0.9523**	0.9439	0.9617
počet epoch=4, LR=$1 \cdot 10^{-5}$, batch size=16	0.9365	0.9408	0.9315	počet epoch=5, LR=$1 \cdot 10^{-4}$, steps_per_epoch=1200	0.9520	0.9516	0.9525
počet epoch=2, LR=$2 \cdot 10^{-5}$, batch size=16	0.9365	0.9360	0.9370	počet epoch=6, LR=$1 \cdot 10^{-4}$, steps_per_epoch=1400	0.9520	0.9427	0.9625
počet epoch=3, LR=$2 \cdot 10^{-5}$, batch size=32	0.9364	0.9373	0.9352	počet epoch=6, LR=$1 \cdot 10^{-4}$, steps_per_epoch=1200	0.9517	0.9493	0.9544
počet epoch=4, LR=$4 \cdot 10^{-5}$, batch size=64	0.9358	0.9351	0.9366	počet epoch=5, LR=$5 \cdot 10^{-5}$, steps_per_epoch=1200	0.9515	0.9490	0.9543
Průměr ze 30 nejlepších experimentů	0.9344	0.9322	0.9372	Průměr ze 30 nejlepších experimentů	0.9497	0.9486	0.953

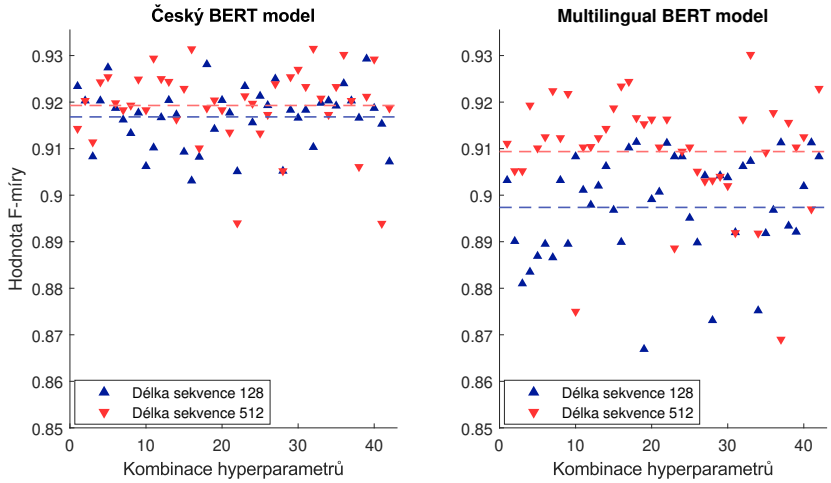
Tabulka 4.3: Nejlepší dosažené výsledky ve smyslu nejvyšší hodnoty F-míry pro modely BERT a T5 na testovacích datech IMDB textového korpusu.

4.1.3.3 Analýza sentimentu ČSFD recenzí

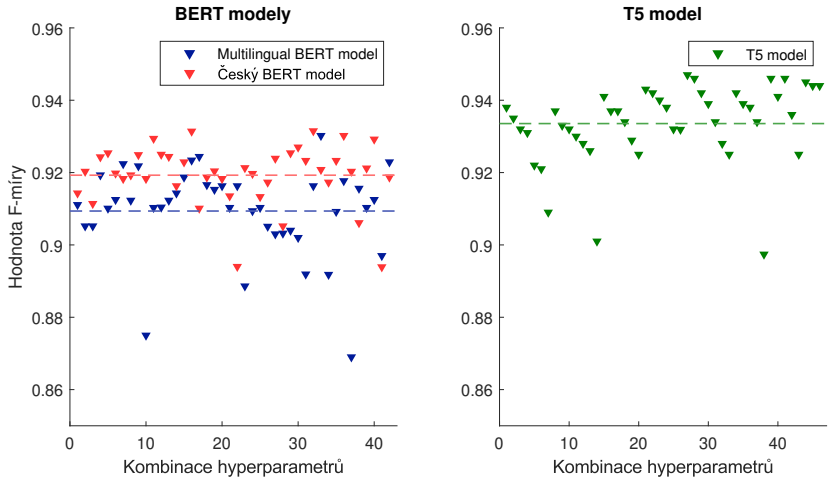
Stejně jako u textového korpusu IMDB byl nejprve testován vliv délky vstupní sekvence BERT modelů na výslednou hodnotu F-míry. Pro některé kombinace hyperparametrů byla výsledná hodnota F-míry přibližně 0.3. Takové kombinace nebyly uvažovány při hodnocení obou modelů. Z grafů 4.6 vyplývá, že změna délky vstupní sekvence má pouze minimální vliv na výkonnost českého BERT modelu pro textový korpus složený z česky psaných recenzí. Pro multilingual BERT model je rozdíl znatelnější, ale stále není tak výrazný jako u korpusu IMDB. Hlavní rozdíl oproti IMDB korpusu je v tom, že průměrná délka recenze pro trénovací dataset je pouze 49 slov a 52 slov pro trénovací dataset (oproti 234 a 229 slovům u IMDB korpusu). Bylo zjištěno, že přibližně 100 recenzí z testovacího korpusu je delších než 128 slov, tzn. že ke zlepšení predikce s největší pravděpodobností dochází právě u těchto vstupních sekvencí (přibližně 1/10 recenzí z testovací množiny). Zároveň lze z grafu vyčíst, že český BERT model výkonností předčil multilingual BERT model pro obě délky sekvencí. Pro další srovnání s T5 modelem byly uvažovány pouze modely s délkou vstupní sekvence 512.

Na obrázku 4.7 je srování BERT modelů s T5 modelem na testovacím datasetu ČSFD recenzí. Z grafů lze opět vyčíst, že obdobně jako u IMDB korpusu i zde

T5 model značně převyšuje BERT modely, co se týká výsledné hodnoty F-míry jednotlivých experimentů.

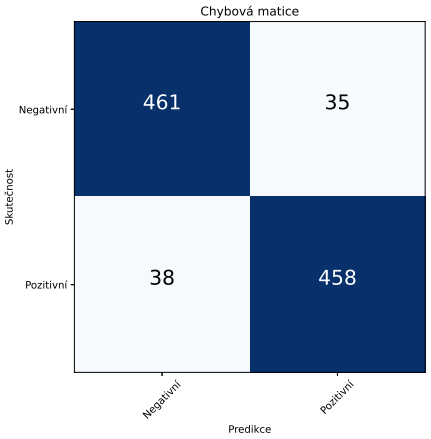


Obrázek 4.6: Vliv délky vstupní sekvence na výslednou hodnotou F-míry. Přerušovaná čára odpovídá střední hodnotě F-míry pro daný model.

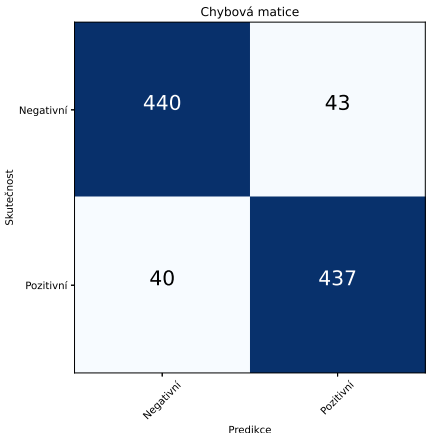


Obrázek 4.7: Srovnání BERT modelů s modelem T5 na ČSFD textovém datasetu. Přerušovaná čára odpovídá střední hodnotě F-míry pro daný model.

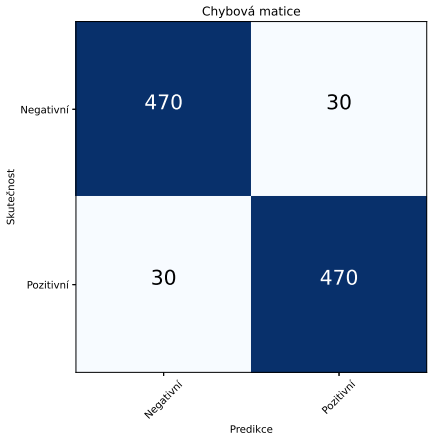
Na obrázku 4.8 jsou zobrazeny tři chybové matice, tj. jedna pro každý testovaný model. Pozitivní recenze jsou obdobně jako u IMDB textového korpusu chybně klasifikovány přibližně stejně jako recenze negativní. V obrázku 4.7 si lze u grafu pro výsledky T5 modelu povšimnout pravidelných vzorů (hodnota F-míry klesá v určitých intervalech). Tento jev je způsoben postupným procházením jednotlivých kombinací hodnot hyperparametrů. V tomto případě docházelo ke změně hodnoty parametru *steps_per_epoch* (od vyšších hodnot k nižším, tj. 1600 až 1000). Je tedy potvrzeno, že ve vyzkoušených kombinacích měl tento hyperparametr velmi velký vliv na výslednou hodnotu F-míry.



Český BERT model
s hyperparametry:
počet epoch: 2, batch size: 32,
LR: $1 \cdot 10^{-5}$.



Multilingual BERT model
s hyperparametry:
počet epoch: 3, batch size: 32,
LR: $1 \cdot 10^{-5}$.



T5 model
s hyperparametry:
počet epoch: 5, steps_per_epoch: 1000,
LR: $5 \cdot 10^{-4}$.

Obrázek 4.8: Srovnání chybových matic všech použitých modelů.

V tabulce 4.4 jsou číselně shrnuty nejlepší dosažené výsledky pro textový korpus ČSFD recenzí. Pro srovnání použitých modelů bylo vybráno 30 nejlepších dosažených výsledků ve smyslu nejvyšší hodnoty F-míry. Získané hodnoty potvrzují tvrzení, že český BERT model dosahuje pro tento textový korpus lepších výsledků než multilingual BERT model. Výsledky tohoto experimentu zároveň potvrzují, že T5 model v úloze analýzy sentimentu výrazně předčil oba BERT modely, a to především průměrnou hodnotou F-míry.

Český BERT model		Multilin. BERT model		T5 model	
Kombinace hyperparametrů	F-míra	Kombinace hyperparametrů	F-míra	Kombinace hyperparametrů	F-míra
počet epoch=3, LR=4e-5, batch size=32	0.9315	počet epoch=3, LR=4e-5, batch size=64	0.9302	počet epoch=6, LR=5e-4, steps_per_epoch=1400	**0.9470**
počet epoch=4, LR=2e-5, batch size=16	0.9314	počet epoch=4, LR=2e-5, batch size=32	0.9244	počet epoch=6, LR=5e-4, steps_per_epoch=1600	0.9460
počet epoch=4, LR=4e-5, batch size=64	0.9302	počet epoch=4, LR=2e-5, batch size=16	0.9234	počet epoch=8, LR=3e-4, steps_per_epoch=1600	0.9460
počet epoch=2, LR=5e-5, batch size=16	0.9294	počet epoch=4, LR=5e-5, batch size=64	0.9229	počet epoch=9, LR=1e-4, steps_per_epoch=1600	0.9460
počet epoch=2, LR=2e-5, batch size=32	0.9294	počet epoch=4, LR=1e-5, batch size=16	0.9224	počet epoch=12, LR=1e-4, steps_per_epoch=1600	0.9450
Průměr ze 30 nejlepších výsledků	0.9232	Průměr ze 30 nejlepších výsledků	0.9153	Průměr ze 30 nejlepších výsledků	0.9393

Tabulka 4.4: Nejlepší dosažené výsledky ve smyslu nejvyšší hodnoty F-míry pro modely BERT multilingual, český BERT a T5 na testovacích datech ČSFD textového korpusu.

Analýza sentimentu je ve většině případů jednodušší úlohou zpracování přirozeného jazyka. Někdy je však velice těžké určit sentiment vybraného textu, viz. následující příklad:

Olmertův um režizérský na motivy umu pisatelského pana Párala, v kombinaci s hereckým umem skvadry typu Kuklové a Pomejeho v celé své kráse...

Tato recenze byla vybrána z ČSFD textového korpusu. Jedná se o recenzi, která byla při jedné z predikcí chybně označena jako pozitivní. Lze však říci, že s klasifikací této recenze by měl každý filmový a divadelní laik obdobné problémy.

4.2 Detekce pojmenovaných entit

Úloha detekce pojmenovaných entit je nedílnou součástí zpracování přirozeného jazyka. Jedná se o jednu z nejtěžších úloh především z důvodu nejednoznačnosti jazyka. Cílem je detekovat a správně zařadit slova nebo slovní výrazy do předem stanovených tříd. Třídy jsou vždy voleny podle účelu použití. Příkladem může být jednoduchá detekce entit typu: *OSOBA, ORGANIZACE, MÍSTO* a mnoho dalších. Lze však detekovat i tzv. vnořené entity. Příklad viz. níže:

Internetový obchod $\overbrace{Amazon}^{\text{ORGANIZACE}}$ je vlastněn $\overbrace{\underbrace{Jeffem}_{\text{jméno}}\ \underbrace{Bezosem}_{\text{příjmení}}}^{\text{OSOBA}}$ pocházejícím

$$\text{ze}\overbrace{\underbrace{\textit{Spojených Států Amerických}}_{\text{stát}}.}^{\text{MÍSTO}}$$

Jak je vidět z předešlého příkladu, u jména lze navíc rozlišovat, zdali se jedná o osobní jméno nebo příjmení. Dále pak u slovního spojení Spojené Státy Americké, které je označené jako místo, lze navíc specifikovat, že se jedná o název státu. Zároveň je z ukázky zřejmé, že entity mohou být víceslovná spojení. Při vnořených entitách však často dochází ke kolizím, například slovní spojení *Most přes řeku Kwai* je název válečného filmu a zároveň se v názvu vyskytuje název řeky *Kwai*. Určení entity vždy závisí na kontextu v jakém je slovní spojení použito.

4.2.1 Dataset CNEC

Prvním datasetem, který byl použit pro detekci entit, se stal korpus CNEC [37], jež byl vytvořen v Ústavu formální a aplikované lingvistiky na Karlově univerzitě.

4.2.1.1 Trénovací data

CNEC korpus byl vydán už ve třech verzích. Poslední verze obsahuje 8993 ručně anotovaných českých vět. Jednotlivé entity jsou klasifikovány do 46 podrobných tříd. Těchto 46 tříd je zároveň shrnuto do 8 hlavních kategorií, viz. tabulka 4.5. V tabulce je pouze 41 detailních entit, a to z toho důvodu, že byly vynechány entity $g\_$, $i\_$, $n\_$, $o\_$, $p\_$. Tyto entity označují blíže nespecifikované detailní entity hlavních kategorií a vyskytují se v trénovacím korpusu zřídka, navíc způsobovaly zbytečné chyby při trénování. Celkově je v korpusu anotováno 35220 pojmenovaných entit. Dochází však k překryvu těchto entit. Takovým příkladem je následující slovní spojení z trénovacího korpusu:

$$\text{<P><pf>Jan</pf><ps>Princ</ps></P>}, \tag{4.4}$$

kde jméno Jan Princ je anotován jako 3 různé entity. Dvě jednoslovné, tj. jméno (<pf>) a příjmení (<ps>), a jedna dvouslovná entita, tj. celé jméno osoby (<P>). V korpusu se navíc nachází i případy, kdy je slovní spojení Ústí nad Labem označeno celé jako město, ale navíc je slovo Labem označeno jako hydronym. Podobné překryvy entit byly ignorovány a byla použita hlavní entita, tj. v tomto případě název města pro celé slovní spojení. Celý korpus je rozdělen v poměru 8:1:1 na trénovací, validační a testovací sadu. Bližší popis entit CNEC korpusu viz. tabulka 4.5.

Nekonzistence ve značení + zajímavosti v korpusu

Při detailnějším zkoumání CNEC textového korpusu byla objevena nekonzistentní značení. V některých větách nejsou označena osobní jména jako dvouslovná

entita typu <P>, ale pouze jako dvě jednoslovné entity typu <pf> a <ps>, tzn. jméno u slovního spojení Jan Princ z ukázky 4.4 by neobsahovalo entitu <P>. Takové značení samozřejmě vnáší chybu do trénovacího korpusu. Navíc pokud se taková chyba ve značení vyskytne i v testovacích datech, dojde ke špatnému vyhodnocení přesnosti modelu jednou z evaluačních technik.

Korpus však neobsahuje pouze celé věty. V textech se například vyskytují kousky televizních a divadelních programů, sportovní výsledky, dokonce několik kontaktů obsahující emailové schránky a telefonní čísla. Jako příklad byl vybrán nevšední text *"země Z S B Z S B celkem SSSR 48 41 19 38 30 30206"*. Takovéto věty poukazují na to, že tento korpus byl stavěn tak, aby co nejlépe reprezentoval reálné texty, avšak častokrát věty bez okolního kontextu pozbývají smysl.

Dalším nedostatkem ve značení entit jsou jednotky spojené s číselnou hodnotou této jednotky. Zde dochází k velké nekonzistenci ve značení, někdy je označena pouze samotná číslovka, někdy číslovka i s jednotkou a někdy pouze jednotka. Taková označení opět zavádí nepřesnost při klasifikaci entit.

Všechny zmíněné nedostatky obsažené v korpusu CNEC byly v rámci této diplomové práce ignorovány. Úprava textového korpusu by byla výrazně nad rámec rozsahu diplomové práce. Jedná se však o zajímavý námět pro další výzkum.

Hlavní kategorie	**Detailní kategorie**
Čísla v adresách (a)	čísla ulice (ah, 144), směrovací čísla (az, 94), telefonní čísla (at, 165)
Geografické názvy (g)	hydronyma (gh, 41), městské části (gq, 126), státy (gc, 814), přírodní oblasti (gl, 79), teritoriální oblasti (gr, 158), ulice,náměstí (gs, 236), kontinenty (gt, 70), města (gu, 1552)
Instituce (i)	konference-soutěže (ia, 173), kult.,vzdělávací,vědecké inst. (ic, 1139), společnosti (if, 677), vládní, politické inst. (io, 602)
Média (m)	internetové odkazy (mi, 16), emailové adresy (me, 68), rádia a TV stanice (ms, 61), časopisy (mn, 154)
Číselné výrazy (n)	věk (na, 77), čísla stránek,kapitol,atd. (nb, 202), čísla položek (ni, 60), základní číslovky (nc, 1531), pořadová čísla (no, 257), sportovní skóre (ns, 99)
Artefakty (o)	kulturní artefakty (knihy, filmy) (oa, 1337), měrné jednotky (oe, 381), měny (om, 142), produkty(op, 306), direktiva, normy (or, 112)
Osobní jména (p)	národnostní jména (pc, 169), akademické tituly (pd, 83), jména (pf, 2368), příjmení (ps, 3073), druhá jména (pm, 80), náboženské osoby (pp, 85)
Časové výrazy (t)	hodiny (th, 1124), dny (td, 411), měsíce (tm, 466), roky (ty, 1032), svátky (tf, 18)

Tabulka 4.5: Rozdělení entit do kategorií. V závorce je uvedena zkratka entity a počet výskytů v trénovacích datech.

4.2.1.2 BERT model

Pro detekci pojmenovaných entit byly vytvořeny hned čtyři různé modely. Dva modely pro detekci 7 hlavních entit[10] a dva modely pro detekci 42 detailních entit.

Modely pro detekci 7 hlavních entit mají stejnou architekturu. Liší se však v použitém předtrénovaném modelu. Pro první pokusy byl stejně jako u úlohy analýzy sentimentu použit volně dostupný multi-jazykový model BERT_{BASE} Multilingual Cased [14] s využitím WordPiece tokenizátoru.

Druhým použitým předtrénovaným modelem byl BERT model předtrénovaný pouze na českém trénovacím korpusu [36]. Oproti BERT_{BASE} modelu byl použit SentencePiece tokenizátor [7].

Aby bylo možné použít tyto modely pro úlohu detekce pojmenovaných entit, bylo potřeba upravit vstupní data a výstupní vrstvy modelu. Detekce entit jako taková na rozdíl od analýzy sentimentu pracuje s jednotlivými slovy a slovními spojeními. Jelikož jsou některá slova rozdeleňa do více tokenů, je potřeba s tímto jevem počítat i při přípravě vstupních dat. Příkladem může být věta *"Příbram je krásné město."*, kde slovo Příbram označuje geografický název (název města). Vstupní sekvence je rozdeleňa WordPiece tokenizátorem následovně:

["P", "##ří", "##bra", "##m", "je", "kr", "##ás", "##né", "město", "."].

Slovo Příbram je rozděleno na čtyři různé tokeny. Je tedy nutné, aby se taková skutečnost promítla i do vstupního vektoru označujícího entity ve vstupní sekvenci:

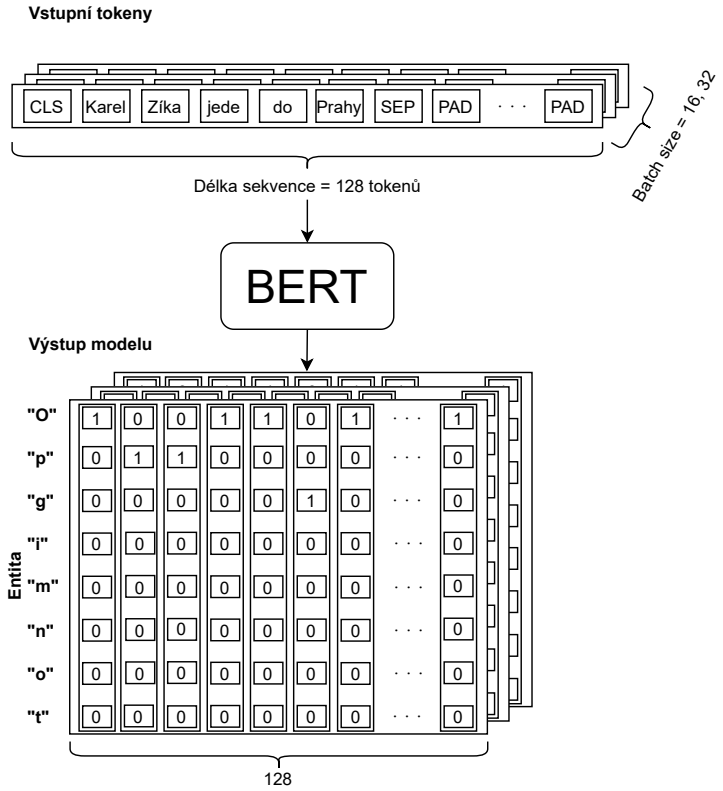
$$\text{["g", "g", "g", "g", "O", "O", "O", "O", "O"]}, \tag{4.5}$$

kde *g* značí entitu *Geografické názvy* a *O* značí nulovou entitu, tzn. že daný token nepatří k žádné hlavní entitě. Velikost vstupních vektorů je fixována na velikost 128, tzn. že zbytek vektoru je doplněn o speciální znaky $[PAD]$.

Výstupem modelu jsou tenzory velikosti $n \times 128 \times 8$,[11] kde n je batch size, viz. obrázek 4.9. To znamená, že pro každý vstupní token je na výstupu k dispozici 8 prvkový vektor obsahující hodnoty od 0 do 1. Suma všech hodnot v takovém vektoru je rovna 1. Jinými slovy je na výstupu vektor, který každé entitě přiřazuje pravděpodobnost právě pro jeden token.

[10] Entity *čísla v adresách* (a) a *číselné výrazy* (n) byly spojeny do jedné entity *číselné výrazy* (n).

[11] (Délka vstupní sekvence) × (počet entit + 1 entita empty). Entita empty slouží pro sekvence, u kterých není detekována žádná hlavní entita.



Obrázek 4.9: Výstup BERT modelu pro detekci 7 hlavních pojmenovaných entit. Jedná se pouze o ilustrační příklad, ve skutečnosti bude většina hodnot nenulová.

Pro detekci 42 detailních (41 z tabulky 4.5 + 1 entita typu P) entit byly použity stejné předtrénované modely jako pro detekci 7 entit, avšak bylo potřeba oproti předešlým modelům upravit architekturu výstupní vrstvy a zároveň s tím i vstupní data. Pro reprezentaci vnořených entit slouží dva vektory, které mají stejnou strukturu jako vektor 4.5. Díky těmto vektorům je možné popsat vnořené entity do hloubky dva, viz. příklad níže:

Vstupní sekvence:

Ing. Josef Blábolil, 67 let, Chomutov.

Vstupní sekvence ve formě tokenů:

[”Ing”, ”.”, ”Josef”, ”B”, ”##lá”, ”##bol”, ”##il”, ”,”, ”67”, ”let”, ”,”, ”Cho”, ”##mut”, ”##ov”].

Vektory označující entity obsažené ve vstupní sekvenci:

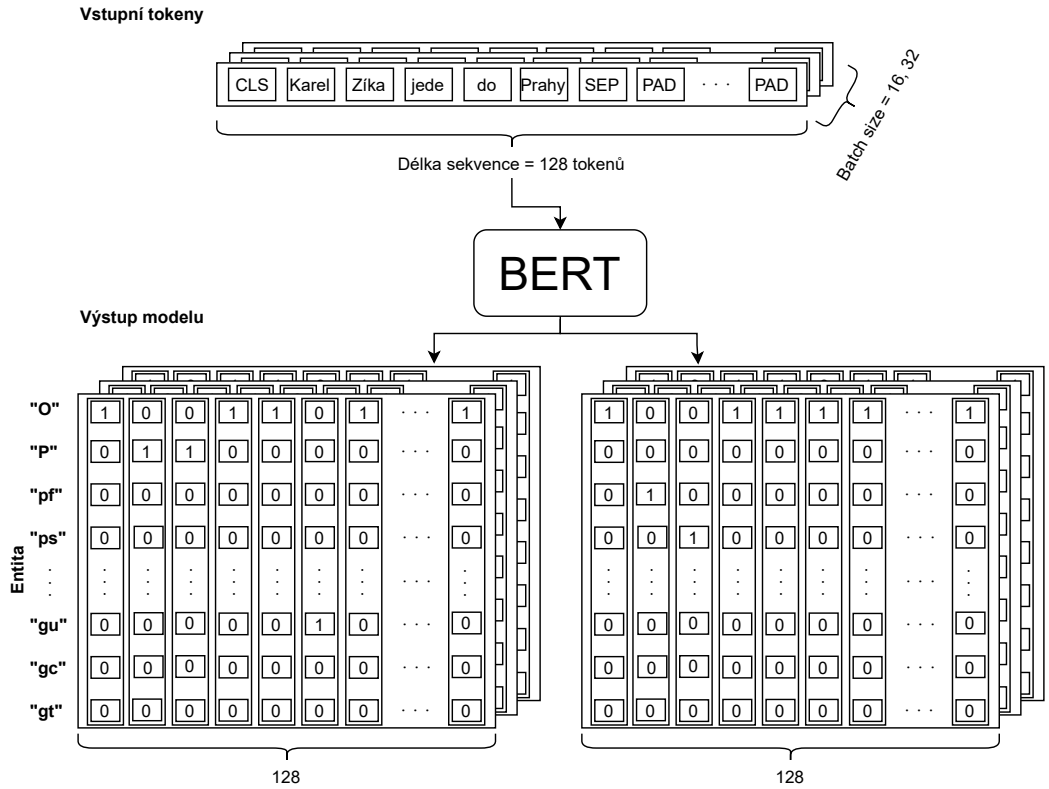
[[”pd”, ”pd”, ”P”, ”P”, ”P”, ”P”, ”P”, ”O”, ”na”, ”O”, ”O”, ”gu”, ”gu”, ”gu”], [”O”, ”O”, ”pf”, ”ps”, ”ps”, ”ps”, ”ps”, ”O”, ”O”, ”O”, ”O”, ”O”, ”O”, ”O”]].

Z příkladu je patrné, že tímto přístupem lze jednoduše reprezentovat vnořené entity, například jméno složené z osobního jména a příjmení. Hloubka dva byla zvolena až po několika testování, kdy bylo zjištěno, že model natrénovaný na trénovacích

datech CNEC korpusu je schopný predikovat právě do takové hloubky.

Se změnou vstupních dat bylo potřeba změnit i výstup modelů. Použita byla architektura *multi-output model*. Tato architektura má hned několik výstupních vrstev. V tomto případě byly použité dvě výstupní vrstvy, které reprezentují jednotlivé hloubky predikce entit. Oba výstupní tenzory jsou velikosti $n \times 128 \times 43$,[11] kde n je batch size, viz. obrázek 4.10.

U všech modelů je pro výpočet ztrátové funkce využito kritéria *categorical cross-entropy*.



Obrázek 4.10: Výstup BERT modelu pro detekci 42 detailních pojmenovaných entit. Každý z výstupů modelu reprezentuje jednu úroveň predikce vnořených entit. Jedná se pouze o ilustrační příklad, ve skutečnosti bude většina hodnot nenulová.

4.2.1.3 T5 model

Pro detekci pojmenovaných entit byl použit totožný předtrénovaný model jako pro analýzu sentimentu ČSFD recenzí. Na rozdíl od BERT modelů nebylo potřeba nijak upravovat architekturu modelu, především pak výstupní vrstvy modelu. Charakter úlohy je definován formou vstupních dat. V tomto případě byla zvolena následující struktura:

Vstupní text:

Město Washington, tvořící vyčleněné území District of Columbia, se stalo hlavním městem USA a sídlem federální vlády v roce 1800.

Očekávaný výstup (42 detailních entit):

<gu> Washington </gu><gr> District of Columbia </gr>
<gc> USA </gc><ty> 1800 </ty>.

Očekávaný výstup (7 hlavních entit):

<g> Washington </g><g> District of Columbia </g>
<g> USA </g><t> 1800 </t>.

Výstupní text obsahuje všechny detekované entity ze vstupního textu. Ukázky výše jsou z trénovacích dat CNEC korpusu. Je patrné, že struktura vstupních dat T5 modelu je na rozdíl od BERT modelu velice jednoduchá a jakékoliv modifikace (viz. změna počtu entit) jsou jednoduše proveditelné a není potřeba měnit architekturu modelu.

4.2.2 Dataset HHTT + TIA

Tématem této diplomové práce je porozumění řeči založené na neuronových sítích, avšak žádný z dosud použitých textových korpusů nevznikl z řečových promluv. Pro analýzu sentimentu byly použity psané recenze a CNEC korpus pro detekci pojmenovaných entit byl vytvořen z vybraných psaných vět. Pro poslední experimenty byl proto vybrán korpus HHTT + TIA, obsahující promluvy, textové přepisy a významové anotace.

4.2.2.1 Trénovací data

Tento dataset vznikl sloučením promluv z HHTT korpusu (Human-Human Train Timetable) [38] a korpusu TIA (Telefonní inteligentní asistentka) do jednoho multi-task korpusu.

HHTT korpus vznikl pro vývoj hlasového dialogového systému v úloze *Nádraží* řešené na Katedře kybernetiky Západočeské univerzity v Plzni. Jedná se o sémanticky anotovaný textový korpus. V korpusu jsou obsaženy záznamy dialogů, které probíhaly při provozu systému poskytujícího informace o příjezdech a odjezdech vlaků. Jelikož se jedná o dialogy typu člověk-člověk, obsahují záznamy dialogu i neřečové události, např. ehm-hmm, které jsou typické pro tento typ komunikace. Anotační schéma korpusu je založeno na konceptu *DATE* - viz. článek *DATE: A Dialogue Act Tagging Scheme for Evaluation of Spoken Dialogue Systems* [39]. V tabulce 4.7 jsou sémantické koncepty použité v korpusu HHTT. V tabulce 4.6 je pak ukázka anotovaného dialogu, ve kterém se uživatel snaží získat informace o odjezdu vlaků. [40]

č.	*ml.*	*promluva*	*sémantický strom*
1	O	informace prosím	GREETING
2	U	dobrý den já bych potřebovala zítra ráno kolem osmé nebo sedmé nějaký vlak do prahy	GREETING DEPARTURE(TIME, TO(STATION))
3	O	takže buď vám jede šest třicet šest rychlík ten je v praze osm deset a nebo rychlík devět čtyřicet šest a praha devět čtyřicet šest	DEPARTURE(TIME, TRAIN_TYPE) ARRIVAL(TO(STATION), TIME) DEPARATURE(TO(STATION), TIME) ARRIVAL(TO(STATION), TIME)
4	U	devět čtyřicet šest	TIME
5	O	ano	ACCEPT
6	U	a všechno jsou to rychlíky	TRAIN_TYPE
7	O	oba dva ano	ACCEPT
8	U	děkuji nashledanou	- -
9	O	není zač nashledanou	- -

Tabulka 4.6: Ukázka anotovaného dialogu z korpusu HHTT. Sloupec *ml.* určuje mluvčího (U – uživatel, O – operátor). [40]

koncept	*popis*	*koncept*	*popis*
ACCEPT	Souhlas	PERSON	Jméno volající osoby
AMOUNT	Cena	PLATFORM	Nástupiště
AREA	Odkud uživatel telefonuje	PREVIOUS	Otázka/odpověď na předchozí spoj
ARRIVAL	Otázka/odpověď pro příjezd	PRICE	Otázka/odpověď na cenu
BACK	Otázka/odpověď pro zpát. spoj	REF	Odkaz na již zmíněné
DELAY	Zpoždění	REJECT	Nesouhlas
DEPARTURE	Otázka/odpověď pro odjezd	REPEAT	Požadavek na zopakování
DISCONNECT	Informace o odpojení vagónů	STATION	Informace o stanici
DISTANCE	Otázka/odpověď na vzdálenost	SYSTEM_FEATURE	Dotaz na služby operátora
DURATION	Doba jízdy	THROUGH	Průjezdní stanice
FROM	Stanice odjezdu	TIME	Čas nebo datum
GREETING	Pozdrav	TO	Cílová stanice
LENGTH	Vzdálenost stanic	TRAIN_TYPE	Typ vlaku
MAYBE	Nejistá odpoveď	TRANSFER	Informace o přestupu
NEXT	Otázka/odpověď na následující spoj	WAIT	Informace o čekání
NUMBER	Číslo reprezentující čas a cenu	WHAT_TIME	Dotaz na čas nebo datum
OTHER_INFO	Jiná informace		

Tabulka 4.7: Sémantické koncepty korpusu HHTT. [40]

Korpus TIA vznikl v rámci výzkumného projektu MPO TIP FR-T11/518 Inteligentní telefonní asistentka řešeného firmou SpeechTech s.r.o. a Katedrou kybernetiky Západočeské univerzity v Plzni. Systém Telefonní inteligentní asistentka vznikl za účelem organizování času, plánování schůzek, rezervace zasedacích místností a mnoho dalšího, a to pouze pomocí hlasových příkazů. Korpus je složen ze dvou částí. První část obsahuje simulované chování budoucího systému. Skládá se ze 187 dialogů a 2469 vět. Druhá část zahrnuje promluvy obsahující vybrané sémantické entity. Jedná se o odpovědi na předem připravené otázky typu *Kdy jste se narodil?* (odpověď obsahuje entitu *datum*), *Svítí dnes slunce?* (odpověď obsahuje entitu *souhlas/nesouhlas*). Druhá část obsahuje 210 dialogů a 3698 vět. Sémantické koncepty tohoto korpusu jsou rozděleny na více podmnožin, tj. *sémantické entity* - např. datum, časy, jména lidí, *sémantické akce* - příkazy, které má hlasový dialogový systém vykonat, *sémantické cíle* - definují funkcionalitu dialogového systému. Jednotlivé sémantické koncepty se nachází v tabulce 4.9. V tabulce 4.8 se nacházejí ukázky promluv z korpusu TIA. [40]

Textový korpus HHTT + TIA obsahuje jak data přepsaná anotátorem, tj. to, co bylo opravdu řečeno, tak i data získaná automatickým rozpoznáváním řeči. Modul rozpoznávání řeči pro jednotlivé korpusy má následující přesnost - HHTT: *ASR Acc = 75.0%*, TIA: *ASR Acc = 77.9%*. To znamená, že přibližně každé čtvrté rozpoznané slovo je rozpoznáno chybně.

promluva	*abstraktní sémantický strom*
potřebovala bych rezervovat zasedací místnost na čtvrtek ve čtyři hodiny	VYTVOR(REZERVACE(VEC, DATUM, T))
je tento termín volný	ZJISTI(KALENDAR)
na zítra v deset hodin zasedací místnost číslo tři	VYTVOR(REZERVACE(DATUM, T, VEC))
ne to stačí	NE
všechny problémy mám vyřešené	OOT
již nemám další dotaz ukončím tuto anketu	NE
jaké schůzky mám sjednaný na p+ sjednané na příští týden	ZJISTI(KALENDAR(RELATIVNI))
ne již nemám žádné přání děkuji	NE, DIKY
ne všechny problémy mám vyřešené	NE

Tabulka 4.8: Ukázkové promluvy z korpusu TIA, symbol + značí nedokončené části promluv.

entita	*popis*
HELLO	Pozdrav + představení
DIKY	Poděkování
BYE	Rozloučení
DATUM	Vyjádření data, nezávislé na aktuálním dni
T	Vyjádření času, včetně předložek
MISTO	Vyjádření místa
JMENO	Jméno osoby
VEC	Název prostředku
SOUHLAS	Souhlas
NE	Nesouhlas
SUBJECT	Název události
INTERVAL	Časový interval
RELATIVNI	Relativní čas nebo datum

akce	*popis*
ZJISTI	Dotaz na objektj
VYTVOR	Vytvoření objektu
ZRUS	Zrušení objektu
UPRAV	Úprava objektu

cíl	*popis*
OOT	Mimo řešenou doménu
KALENDAR	Osobní kalendář
SCHUZKY	Skupinové plánování
REZERVACE	Rezervace prostředků
SPOJ	Zprostředkování hovoru
KONFERENCE	Konferenční hovory
POZNAMKA	Čtení, uložení poznámek

Tabulka 4.9: Sémantické entity, sémantické akce a sémantické cíle korpusu TIA.

4.2.2.2 T5 model

Pro úlohu detekce sémantických entit v korpusu HHTT + TIA byl použit pouze model T5 shodný s modelem pro detekci pojmenovaných entit CNEC korpusu. Cílem bylo zjistit, zdali je předtrénovaný T5 model konkurence schopný pro model, který byl navržen přímo pro danou úlohu, tj. konvoluční neuronová síť trénovaná z rozpoznaných mřížek.

Celý dataset byl konvertován do šesti *tsv* souborů, kde na každém řádku je promluva a neuspořádaný sémantický strom odpovídající dané promluvě. Rozdělení dat je znázorněno v tabulce 4.10. Rozdíly v počtech vět přepsaných anotátorem a vět získaných automatickým rozpoznáváním řeči jsou způsobeny chybovostí rozpoznávače.

Data byla v následujícím tvaru:

Vstupní text:

anotovaný přepis: *_nse_ prosím _nse_ raději ne protože jsme si neupřesnili čas*
data z rozpoznávače: *ano prosím na raději ne protože jsme si neupřesňuji čas*

Očekávaný výstup:

REJECT, ZRUS

Zkratka *_nse_* v anotovaném přepisu značí neřečové události v promluvě. Dále je patrné, že v rozeznané promluvě došlo k chybám. Bylo rozpoznáno slovo *ano*, které však v promluvě vysloveno nebylo a zároveň místo slova *neupřesnili* bylo detekováno slovo *neupřesňuji*.

	Počet vět		
	Trénovací data	Validační data	Testovací data
Data přepsaná anotátorem	11665	1089	2695
Data získaná automatickým rozpoznáváním řeči	11607	1086	2688

Tabulka 4.10: Počty vět v jednotlivých tsv souborech datasetu HHTT + TIA.

4.2.3 Výsledky

BERT modely byly testovány s několika kombinacemi hyperparametrů. Oproti analýze sentimentu bylo zapotřebí použití většího počtu trénovacích epoch. To je způsobeno především větší složitostí této úlohy. Doba trénování takových modelů se výrazně prodloužila. Byly testovány následující hodnoty hyperparametrů (celkem 80 kombinací):

- počet epoch: 10, 12, 14, 16, 18, 22, 28, 32
- learning rate: $1 \cdot 10^{-5}$, $2 \cdot 10^{-5}$, $3 \cdot 10^{-5}$, $4 \cdot 10^{-5}$, $5 \cdot 10^{-5}$
- batch size: 16, 32
- délka sekvence: 128

Obdobně jako u předešlých experimentů byl výběr vhodných hyperparametrů pro T5 model proveden experimentálně, kombinací různých hodnot hyperparametrů.

4.2.3.1 Evaluační metriky

Pro následující zhodnocení výsledků není možné použít obyčejné metriky použité v úloze analýzy sentimentu. Metriky je potřeba upravit pro detekci vnořených entit a určit jakým způsobem bude daná metrika penalizovat chybnou detekci. U CNEC korpusu se vnořené entity zbaví tagů a v potaz se berou pouze listy stromu ve formátu XML. Jako příklad opět poslouží slovní spojení Jan Princ:

Reference:

<P><pf> Jan </pf><ps> Princ </ps></P>

Hypotéza:

<P> Jan <ps> Princ </ps></P>

Pro předchozí hypotézu budou následující entity:

- falešně negativní (FN):
  - hypotéza - Jan
  - reference - <pf> Jan </pf>
- skutečně pozitivní (TP):
  - hypotéza - <ps> Princ </ps>
  - reference - <ps> Princ </ps>
- skutečně pozitivní (TP):
  - hypotéza - <P> Jan Princ </P>
  - reference - <P> Jan Princ </P>

Celkem se jedná o 3 entity. Entita <pf> byla chybně nedetekována a entity <P>, <ps> byly detekovány správně. Pro tento příklad by výsledné hodnoty přesnosti, úplnosti a F-míry byly následovné:

$$\text{Přesnost} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{2}{2+0} = 1 \tag{4.6}$$

$$\text{Úplnost} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{2}{2+1} = \frac{2}{3} \approx 0.66 \tag{4.7}$$

$$\text{F-míra} = 2 \cdot \frac{\text{Přesnost} \cdot \text{Úplnost}}{\text{Přesnost} + \text{Úplnost}} = 2 \cdot \frac{1 \cdot \frac{2}{3}}{1 + \frac{2}{3}} = \frac{4}{5} = 0.8 \tag{4.8}$$

Stav *falešně pozitivní* (FP) by nastal v případě, že by se v hypotéze vyskytla entita navíc, tzn. entita, která se nevyskytla v referenci - například pokud by byla predikce <P><pd> Ing. </pd> Jan <ps> Princ </ps></P> a reference stejná jako v předešlém případě, pak entita <pd> by byla označena jako falešně pozitivní.

Pro detekci sémantických entit byla použita metrika SAcc, jež určuje procento správně predikovaných sémantických stromů.

4.2.3.2 Detekce pojmenovaných entit pro CNEC textový korpus

Aby bylo možné srovnat modely BERT a T5, bylo potřeba sjednotit i výstupní data těchto architektur. Byla zvolena struktura použitá pro očekávaný výstup T5 modelu, viz. část 4.2.1.3. BERT model však na výstupu poskytuje informaci o entitách na úrovni tokenů, a byla tedy zapotřebí složitější úprava výstupních dat než v případě T5 modelu, který generuje výstup rovnou v očekávané podobě. U BERT

modelu navíc dochází k jevu, kdy slovo složené z více tokenů obsahuje více odlišných entit:

Vstupní tokeny:

["P", "##ří", "##bra", "##m"].

Detekované entity na úrovni tokenů:

["O", "gu", "gu", "gu"].

V tomto případě nebyly tokeny jednoho slova označeny stejnou entitou. Pro odstranění takového jevu byla vytvořena jednoduchá metoda, která seskupí tokeny zpět do slov a vybere entitu, jež se vyskytuje v označení tokenů nejčastěji. Tento přístup lze jednoduše použít i pro vnořené entity. K takové chybě dochází především u multilingual modelu, jehož slovník neobsahuje tolik českých slov jako slovník českého BERT modelu, tzn. neviděná slova jsou nahrazena kombinací více tokenů. A právě při použití více tokenů pro jedno slovo dochází k chybné detekci entit na úrovni tokenů.

T5 model je schopný generovat požadovaný výstup rovnou v textové podobě, avšak přibližně 3% predikcí obsahují data v chybném formátu, viz. následující ukázka: Vstupní text:

Internetový obchod Globtrotero , Havlíčkova 7 , 541 01 Trutnov , Česká republika, tel.: +420 607492150 , Provozovatel: Marek Bejr , IČ: 73995711 , kamenná prodejna: Globtrotero , Štefánikova 976 , 530 02 Pardubice , Česká republika

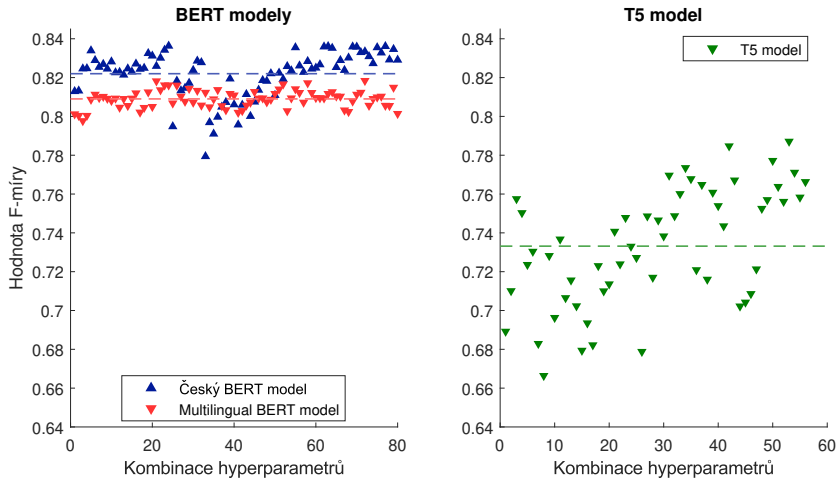
Textový výstup:

Globtrotero <g>Havlíčkova </g><n>7 </n><n>541 01 </n> <g>Trutnov </g><g>Česká republika </g><n>+420 607492150 </n> Marek Bejr <n>73995711 </n>Globtrotero <g>Štefánikova Globtrotero </g></g></n></n> </n></n></n></g></g></g>+420 607492150 </n></n>

Je zřetelné, že v druhé polovině textového výstupu dochází k narušení struktury potřebné k evaluaci predikce, tzn. tagy nejsou ve formě *<název entity>text </název entity>*. Některé predikce T5 modelu dokonce obsahovaly neexistující entity. K tomuto nedostatku dochází u detekce 7 hlavních i 42 detailních entit. Problém nekorektní struktury výstupních dat byl odstraněn jednoduchou úpravou, kdy byly uvažovány pouze validně vygenerované tagy označující entity.

Hodnoty F-míry získané při experimentech se 7 hlavními, resp. 42 detailními entitami, jsou zobrazeny v grafu 4.11, resp. v grafu 4.12. V grafu 4.12 si lze povšimnout, že u výsledných hodnot je opět patrný pravidelný vzor. Při zvyšujícím se počtu epoch se zvyšuje i výsledná hodnota F-míry. Z tabulek 4.11 a 4.12 je zároveň patrné, že při detekci 42 entit je u všech modelů potřeba mnohem více trénovacích epoch než při detekci 7 entit. Ze všech získaných výsledků bylo vybráno 30 nejlepších ve smyslu nejvyšší hodnoty F-míry pro každý model. V obou experimentech

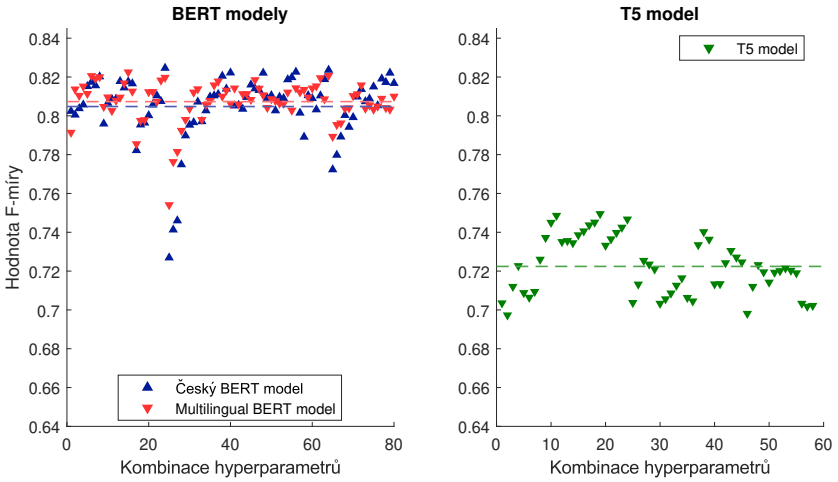
T5 model svými výsledky lehce zaostává za výsledky BERT modelů. To může být způsobeno zvyšující se náročností úlohy. Textový výstup T5 modelu už není pouze jedno slovo (jako tomu bylo u analýzy sentimentu), ale jedná se o složitou strukturu popisující entity ve vstupním textu. Výsledná data viz. tabulky 4.11 a 4.12.



Obrázek 4.11: Srovnání BERT a T5 modelu pro detekci 7 hlavních entit CNEC korpusu. Přerušovaná čára odpovídá střední hodnotě F-míry pro daný model.

Český BERT model		Multilin. BERT model		T5 model	
Kombinace hyperparametrů	F-míra	Kombinace hyperparametrů	F-míra	Kombinace hyperparametrů	F-míra
počet epoch=32, LR=5e-5, batch size=32	**0.8363**	počet epoch=32, LR=4e-5, batch size=16	0.8184	počet epoch=6, LR=5e-4, steps_per_epoch=1400	0.7871
počet epoch=18, LR=5e-5, batch size=16	0.8361	počet epoch=18, LR=3e-5, batch size=32	0.8182	počet epoch=15, LR=5e-4, steps_per_epoch=1700	0.7847
počet epoch=28, LR=3e-5, batch size=16	0.8361	počet epoch=12, LR=3e-5, batch size=16	0.8172	počet epoch=15, LR=5e-4, steps_per_epoch=700	0.7772
počet epoch=18, LR=4e-5, batch size=16	0.8359	počet epoch=16, LR=2e-5, batch size=16	0.8166	počet epoch=9, LR=5e-4, steps_per_epoch=1200	0.7735
počet epoch=22, LR=3e-5, batch size=16	0.8359	počet epoch=28, LR=3e-5, batch size=32	0.8161	počet epoch=7, LR=5e-4, steps_per_epoch=1400	0.7711
Průměr ze 30 nejlepších výsledků	0.8321	Průměr ze 30 nejlepších výsledků	0.8135	Průměr ze 30 nejlepších výsledků	0.7582

Tabulka 4.11: Nejlepší dosažené výsledky detekce 7 hlavních entit ve smyslu nejvyšší hodnoty F-míry pro modely BERT multilingual, český BERT a T5. Výsledky byly získány na testovacích datech CNEC textového korpusu.



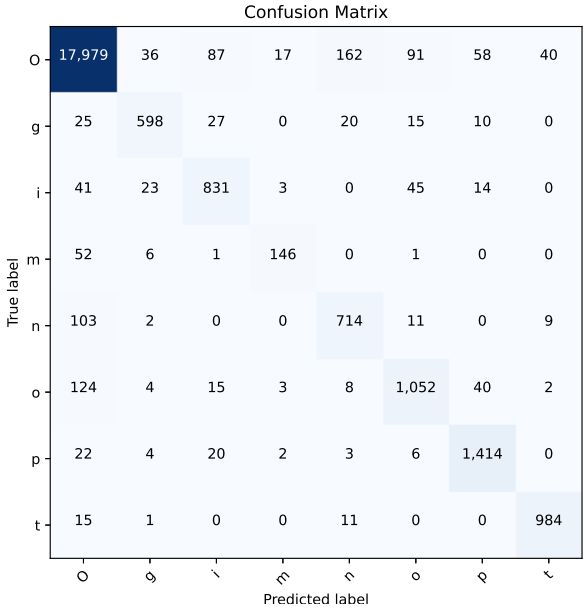
Obrázek 4.12: Srovnání BERT a T5 modelu pro detekci 42 detailních entit CNEC korpusu. Přerušovaná čára odpovídá střední hodnotě F-míry pro daný model.

Český BERT model		Multilin. BERT model		T5 model	
Kombinace hyperparametrů	F-míra	Kombinace hyperparametrů	F-míra	Kombinace hyperparametrů	F-míra
počet epoch=32, LR=2e-5, batch size=32	**0.8246**	počet epoch=28, LR=4e-5, batch size=32	0.8225	počet epoch=30, LR=3e-4, steps_per_epoch=2400	0.7495
počet epoch=32, LR=2e-5, batch size=16	0.8236	počet epoch=32, LR=2e-5, batch size=16	0.8207	počet epoch=30, LR=5e-4, steps_per_epoch=2000	0.7486
počet epoch=32, LR=4e-5, batch size=16	0.8226	počet epoch=22, LR=3e-5, batch size=32	0.8206	počet epoch=30, LR=1e-3, steps_per_epoch=2400	0.7467
počet epoch=32, LR=5e-5, batch size=32	0.8222	počet epoch=32, LR=3e-5, batch size=32	0.8200	počet epoch=25, LR=3e-4, steps_per_epoch=2400	0.7451
počet epoch=28, LR=5e-5, batch size=16	0.8221	počet epoch=32, LR=2e-5, batch size=32	0.8197	počet epoch=20, LR=5e-4, steps_per_epoch=2000	0.7450
Průměr ze 30 nejlepších výsledků	0.8174	Průměr ze 30 nejlepších výsledků	0.8157	Průměr ze 30 nejlepších výsledků	0.7344

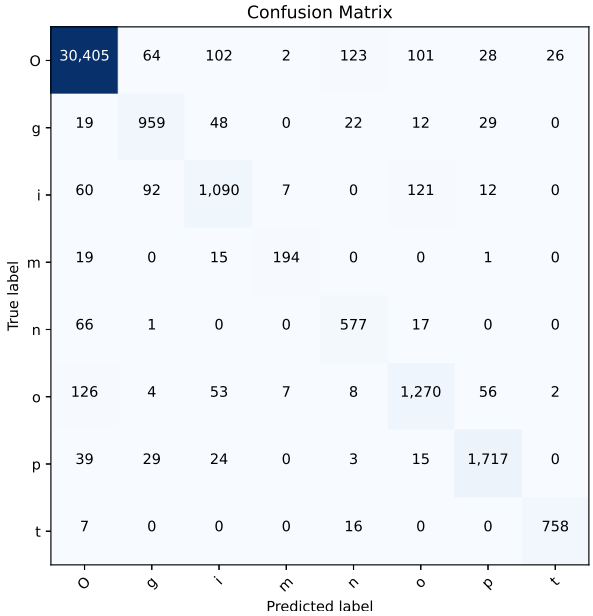
Tabulka 4.12: Nejlepší dosažené výsledky detekce 42 detailních entit ve smyslu nejvyšší hodnoty F-míry pro modely BERT multilingual, český BERT a T5. Výsledky byly získány na testovacích datech CNEC textového korpusu.

Na obrázku 4.13 jsou zobrazeny chybové matice nejlepšího dosaženého výsledku českého a multilingual BERT modelu. Pro T5 model nebyla taková matice vytvořena, protože z výstupu T5 modelu nelze jednoduše určit, jaká entita byla zaměněna za jinou. Z chybových matic je například patrné, že pokud je entita typu

časové výrazy špatně určena, pak je většinou zaměněna za entitu typu *číselné výrazy* (příkladem je slovní spojení *16 h*, kde je číslo 16 označeno jako číselný výraz).



(a) Český BERT model.



(b) Multilingual BERT model.

Obrázek 4.13: Chybové matice českého BERT modelu a multilingual BERT modelu. Vytvořeno z nejlepších výsledků modelů ve smyslu nejvyšší hodnoty F-míry. Tyto matice byly vytvořeny na úrovni tokenů. Jelikož oba modely používají jiné slovníky, jsou i sumy jednotlivých hodnot v matici rozdílné.

Chybové matice jsou pouze informativní. I přesto, že byla entita tokenu označena správně, může být konečné slovní spojení nekompletní a v celkovém hodnocení vyhodnoceno jako nezdařilá predikce. Takový jev nastává především u multilingual BERT modelu, a to u entity typu *média*:

Reference:

<m> sales(at)aeropartner.cz </m>

Hypotéza:

<m> at)aeropartner.cz </m>

Z příkladu výše je patrné, že většina tokenů, ze kterých se hypotéza skládá, byla označena správně, avšak v následném vyhodnocení je entita v hypotéze označena jako nekompletní.

		Český BERT model	Multilin. BERT model	T5 model
Entita	empty	0.9328	0.9560	0.9663
	g	0.8430	0.8059	0.7887
	i	0.7930	0.7203	0.7049
	m	0.6747	0.7000	0.6452
	n	0.7509	0.7124	0.7125
	o	0.7251	0.7208	0.6551
	p	0.8954	0.8920	0.8540
	t	0.9162	0.9031	0.8195
F-míra celkem		0.8363	0.8184	0.7871

Tabulka 4.13: Průměrné hodnoty F-míry pro jednotlivé entity. Hodnoty jsou pro nejlepší dosažené výsledky jednotlivých modelů.

V tabulce 4.14 jsou srovnány průměrné hodnoty jednotlivých metrik pro detekci 7, reps. 42 entit. Je patrné, že snížení počtu entit ze 42 na 7 má vliv na výsledné hodnoty metrik. U českého BERT modelu a T5 modelu dochází při použití 7 hlavních entit ke zlepšení průměrné hodnoty F-míry. Naopak u multilingual BERT modelu dochází k mírnému zhoršení. Snížení počtu entit však neznamená nižší počet entit v textu. Jejich počet je zachován. Příkladem snížení počtu entit jsou entity typu *pf, ps, pm, pd, pc, pp*, které jsou jednotně označeny jako entita typu *p*. Z detailnějšího zkoumání bylo zjištěno, že modely nejčastěji chybují u entit typu *média, číselné výrazy* a *artefakty*. Naopak nejlépe jsou klasifikovány entity typu *osobní jména, časové výrazy* a *geografické názvy* (viz. tabulka 4.13). Tento jev byl předpokládaný, jelikož se jedná o vcelku jednoznačné a přesně definované typy entit.

Zároveň si lze povšimnout, že při použití 42 entit má T5 model poměrně vysokou hodnotu přesnosti, ale zároveň velmi nízkou hodnotu úplnosti. Jinými slovy lze říci, že T5 model s nižší pravděpodobností detekuje všechny entity v textu, ale pokud nějakou entitu označí, tak většinou správně. Tento jev může být způsoben generováním nevalidních výstupů zmíněných výše. Tyto výstupy jsou následně odstraněny a označeny jako nedetekované.

Hodnoty v tabulce 4.14 potvrzují, že v obou experimentech (detekce 7 a 42 entit) si nejlépe vedl český BERT model.

		Český BERT model		Multilin. BERT model		T5 model	
		Počet entit					
		7	42	7	42	7	42
Průměrná hodnota	F1-míra	0.8321	0.8174	0.8135	0.8157	0.7582	0.7344
	Přesnost	0.8253	0.8082	0.8089	0.8159	0.7642	0.7887
	Úplnost	0.8393	0.8277	0.8196	0.8174	0.7579	0.6897

Tabulka 4.14: Průměrné hodnoty F1-míry, přesnosti a úplnosti pro detekci entit.

Úloha detekce pojmenovaných entit je oproti analýze sentimentu výrazně složitější úlohou. Zároveň se modely potýkaly s nekonzistentním značením, viz. následující ukázka:

Maďarsko požádá OSN o náhradu škod, které mu vzniknou zavedením ekonomických sankcí vůči Jugoslávii.

Pro tuto větu byla v trénovacích datech CNEC korpusu dostupná následující reference:

<io> OSN </io><gc> Jugoslávii </gc>

Je patrné, že slovo Maďarsko označující entitu typu <gc> nebylo v referenci označeno. V tomto případě byl například český BERT, který ve stejné větě detekoval následující entity:

<gc> Maďarsko </gc><io> OSN </io><gc> Jugoslávii </gc>,

chybně penalizován. I tak lze však říci, že všechny modely dosáhly velmi dobrých výsledků.

4.2.3.3 Detekce sémantických entit pro HHTT + TIA korpus

Cílem těchto experimentů bylo zjistit, jakých výsledků lze dosáhnout s použitím předtrénovaného T5 modelu na validačních a testovacích datech HHTT + TIA korpusu.

Pro první experiment byl T5 model dotrénován na trénovacích datech obsahující pouze anotované promluvy, tzn. to, co bylo opravdu řečeno (TRNS). Takový model byl evaluován na testovacích datech obsahujících taktéž pouze anotované přepisy promluv. Tím byly získány baseline výsledky, které následně sloužily v dalších experimentech jako referenční hodnoty.

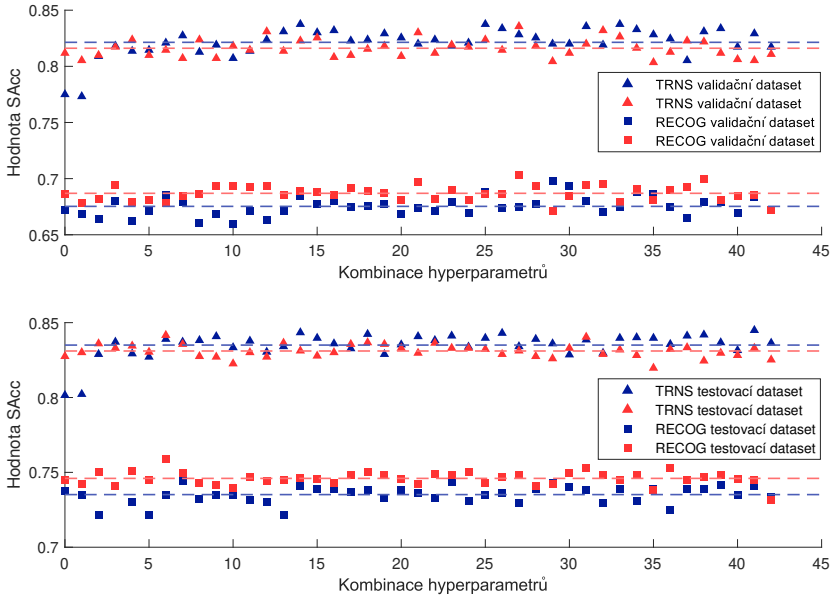
Výkonnost T5 modelu byla demonstrována při experimentu, ve kterém byl T5 model srovnán s modelem navrženým speciálně pro úlohu detekce sémantických entit v HHTT+TIA korpusu. Srovnány byly hodnoty získané na validačních a testovacích datech obsahujících pouze data z rozpoznávače řeči (RECOG). Jelikož průměrná přesnost rozpoznávače je přibližně 75%, lze říci, že průměrně každé čtvrté slovo

je rozpoznáno chybně. To velmi ztěžuje následné určení sémantických stromů a je důležité zjistit, jaký vliv tento jev má na výslednou hodnotu přesnosti.

Následně bylo testováno, zdali má na výsledné hodnoty přesnosti vliv rozšíření trénovacího datasetu o data obsahující rozpoznané promluvy, tzn. model byl dotrénován na anotovaných datech i na datech získaných z rozpoznávače řeči. Výsledky ze všech provedených experimentů jsou zobrazeny v tabulce 4.15

	Data použitá pro dotrénování modelu	
	Trénovací data TRNS	Trénovací data TRNS + trénovací data RECOG
Validační data: TRNS	0.8237	0.8145
Testovací data: TRNS	0.8371	0.8312
Validační data: RECOG	0.6749	0.6860
Testovací data: RECOG	0.7362	0.7458

Tabulka 4.15: Průměrné hodnoty přesnosti T5 modelu. TRNS značí dataset obsahující anotované přepisy promluv a RECOG značí dateset obsahující rozpoznané promluvy.



Obrázek 4.14: Červeně označené výsledky jsou získány při použití pouze TRNS datasetu pro dotrénování modelu. Modře označené výsledky jsou získány při použití rozšířeného datasetu TRNS + RECOG. Přerušované čáry jsou průměrné hodnoty přesnosti pro jednotlivé datasety.

Z tabulky 4.15 a obrázku 4.14 vyplývá, že při evaluaci modelu na datech obsahujících pouze rozpoznané promluvy dochází ke snížení přesnosti modelu. To je způsobeno již zmíněným jevem, kdy v průměru každé čtvrté slovo je rozpoznáno chybně. Dále lze říci, že pokud je model dotrénován na rozšířených trénovacích datech (TRNS + RECOG), pak dochází k mírnému zlepšení při predikci sémantických stromů z rozpoznaných promluv (RECOG datasety). Takové zlepšení je však doprovázeno zhoršením výsledných hodnot přesnosti pro anotované promluvy (TRNS datasety). Zároveň je z grafů zřejmé, že při této úloze nemají tak velký vliv jednotlivé hodnoty hyperparametrů, jako tomu bylo u předešlého použití T5 modelu. Výsledné hodnoty jsou ovlivněny především lehkým šumem.

Pro ukázku vlivu chybovosti rozpoznávače na detekci sémantických entit byla vybrána následující věta:

Anotovaný přepis promluvy:

a z prahy do plzně v pátek někdy večer řekněme tak od sedmi hodin

Rozpoznaná promluva:

aby tady do plzně v pátek někdy večer řekněme tak od sedmi hodin

Je patrné, že kvůli chybně rozpoznané promluvě se částečně mění i význam požadavku. To se podepsalo i na následné detekci sémantických entit, viz. níže.

Reference:

DEPARTURE(FROM(STATION), TO(STATION), TIME, TIME)

Predikce:

TO(STATION), TIME

V porovnání s modelem, který byl navržen přímo pro danou úlohu, dosáhl T5 model obdobných výsledků (viz. tabulka 4.16). Předtrénovaný T5 model narozdíl od navrženého modelu není primárně určen pouze k jedné úloze. Navíc se jedná o základní verzi této architektury bez jakýchkoliv úprav. To poukazuje na slibný potenciál tohoto modelu.

	Validační dataset	Testovací dataset
Referenční model	0.6970	0.7451
T5 model	0.6970	0.7425

Tabulka 4.16: Nejlepší dosažené hodnoty přesnosti pro úlohu detekce sémantických entit. Pro srovnání byla použita metrika SAcc.

4.3 Zhodnocení výsledků

V rámci této práce byl srovnán T5 model s modelem typu BERT, který doposud dosahoval state-of-the-art výsledků v úlohách zpracování přirozeného jazyka. S těmito modely bylo provedeno několik experimentů. Úloha analýzy sentimentu byla testována na českém a anglickém datasetu obsahující filmové recenze. Pro oba jazyky dosáhl T5 model lepších výsledků ve smyslu nejvyšší hodnoty F-míry i ve smyslu průměrné hodnoty F-míry než modely BERT.

Pro úlohu detekce pojmenovaných entit byl použit korpus CNEC, jež je složen z vět obsahujících pojmenované entity. V této úloze T5 model lehce zaostal za BERT modely. A to jak při detekci 7 hlavních pojmenovaných entit, tak i 42 detailních entit. To může být způsobeno především složitostí úlohy. Při analýze sentimentu bylo výstupem T5 modelu pouhé jedno slovo označující třídu sentimentu, kdežto při detekci entit je většinou na výstupu text obsahující i několik označených entit (slov, slovních spojení) najednou.

Při detekci sémantických entit v korpusu HHTT+TIA byl T5 model srovnán s modelem navrženým přímo pro danou úlohu. V těchto experimentech dosáhl T5 model velmi dobrých a srovnatelných výsledků oproti porovnávanému modelu.

Velkou výhodou T5 modelu je výrazně jednodušší příprava vstupních dat. Například pro detekci entit není potřeba mít na vstupu pro každou sekvenci tokenů odpovídající sekvenci entit. U T5 modelu se navíc používá stejná architektura modelu pro různé úlohy zpracování přirozeného jazyka. Úloha je specifikována pouze strukturou očekávaných výstupních dat. U BERT modelu je tomu naopak a je potřeba upravit výstupní vrstvy speciálně pro danou úlohu.

Z provedených experimentů lze vyvodit, že T5 model svou architekturou přináší nové poznatky do zpracování přirozeného jazyka i porozumění mluvené řeči. Ve srovnání s BERT modelem dosáhl velmi slibných výsledků, a to především v úloze analýzy sentimentu. Při detekci pojmenovaných entit T5 model lehce zaostával, avšak stále se jednalo o velmi dobrý výsledek, především ve smyslu nejvyšší dosažené hodnoty F-míry. Průměrná hodnota už byla o něco nižší. To však může být způsobeno experimentálním výběrem hodnot hyperparametrů. Ne vždy se jedná o ideální nastavení, a výsledná hodnota F-míry je proto výrazně nižší oproti nejvyšší dosažené hodnotě. V posledních experimentech se T5 model následně svými výsledky vyrovnal modelu speciálně navrženému pro detekci sémantických entit HHTT+TIA korpusu. Dosažené výsledky poukazují na možný potenciál této nové architektury a je možné, že například při delším běhu procesu předtrénování T5 modelu by tyto výsledky mohly být ještě lepší. To je námět pro další výzkum této architektury.

Výsledky všech provedených experimentů jsou obsahem přílohy této diplomové práce.

5 Závěr

Cílem této práce bylo seznámení s problematikou neuronových sítí v oblasti porozumění mluvené řeči. Především pak se zaměřením na techniky nazývané transfer-learning. Této technice byl věnován velký díl teoretické části této práce. V úvodu byly nastíněny přístupy, které předcházely aktuálním modelům, jež se používají v oblasti zpracování přirozeného jazyka. Následně byla detailně rozebrána architektura transformer modelů. Částečně se tato práce zmínila o vektorové reprezentaci slov. Především pak o tzv. Word2Vec modelu.

Značný úsek teoretické části byl věnován specifickým transformer modelům typu BERT a T5. Tyto architektury se v posledních letech staly state-of-the-art modely v úlohách porozumění přirozenému jazyku. Podrobně byly popsány základní principy, architektury, korpusy použité pro předtrénování a následné použití pro specifické úlohy.

Závěr teoretické části přiblížil principy TensorFlow a PyTorch frameworků, jež byly při experimentech použity.

T5 architektura je novinkou v oblasti zpracování přirozeného jazyka. Jedním z cílů této diplomové práce bylo zjistit, do jaké míry je T5 model schopný konkurovat ostatním state-of-the-art modelům, především pak architektuře typu BERT. Nejprve byly tyto dva modely použity pro analýzu sentimentu anglických a českých filmových recenzí. V práci byly podrobně rozebrány jednotlivé výsledky. Prostudován byl také vliv délky sekvence na výslednou hodnotu F-míry. V úloze analýzy sentimentu dosáhl T5 model výrazně lepších výsledků ve všech experimentech.

Druhou zkoumanou úlohou byla detekce pojmenovaných entit v textovém korpusu CNEC. Jedná se o jednu z nejnáročnějších úloh zpracování přirozeného jazyka, a to především z důvodu nejednoznačnosti jazyka. Nejprve byl prostudován vliv počtu entit na výslednou hodnotu F-míry. Následně byly detailně rozebrány jednotlivé výsledky obou modelů. Diplomová práce se také věnovala nedostatkům modelů při detekci entit. V porovnání s BERT modelem už v této úloze T5 model lehce zaostal. Bylo rovněž prokázáno, že pro úlohy zpracování českého jazyka je vhodnější použít BERT model předtrénovaný na českém textovém korpusu. Tento BERT model v porovnání s multilingual modelem dosáhl lepších výsledků.

Při posledních experimentech byl využit HHTT+TIA korpus obsahující promluvy, textové přepisy a významové anotace. Jednalo se o úlohu detekce sémantických entit. Při těchto experimentech byl T5 model srovnán s modelem speciálně navrženým pro danou úlohu. Výsledné hodnoty přesnosti obou modelů byly srovnatelné, což poukazuje na potenciál T5 modelu.

Provedením a následnou analýzou zmíněných experimentů byly splněny všechny

body zadání této diplomové práce. Vzhledem k tomu, že doposud nebyl vydán žádný článek, který by se věnoval použití T5 modelu pro úlohy zpracování českého jazyka, lze tuto práci považovat za inovativní. Práce se snaží přispět k obohacení dosavadních poznatků a zároveň poskytuje námět k dalším experimentům s touto architekturou.

Literatura

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention Is All You Need*, 2017.

[2] Rahul Agarwal. *What Are Transformer Models in Machine Learning?* https://lionbridge.ai/articles/what-are-transformer-models-in-machine-learning, 2020.

[3] Gabriel Loye. *Attention Mechanism.* https://blog.floydhub.com/attention-mechanism, 2019.

[4] Google Brain Team. *Tensor2Tensor Colab.* https://colab.research.google.com/github/tensorflow/tensor2tensor/blob/master/tensor2tensor/notebooks/hello_t2t.ipynb?authuser=2#scrollTo=OJKU36QAfqOC.

[5] Jay Alammar. *The Illustrated Transformer.* http://jalammar.github.io/illustrated-transformer/, 2018.

[6] Lilian Weng. *Attention? Attention!* https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html#a-family-of-attention-mechanisms, 2018.

[7] Taku Kudo and John Richardson. *SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing.* https://www.aclweb.org/anthology/D18-2012, 2018.

[8] Joao Schapke. *Evolution of word representations in NLP.* https://towardsdatascience.com/evolution-of-word-representations-in-nlp-d4483fe23e93, 2019.

[9] Jayesh Bapu Ahire. *Introduction to Word Vectors.* https://medium.com/@jayeshbahire/introduction-to-word-vectors-ea1d4e4b84bf, 2018.

[10] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. *Efficient Estimation of Word Representations in Vector Space*, 2013.

[11] Dhruvil Karani. *Introduction to Word Embedding and Word2Vec.* https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa, 2018.

[12] Joerg Landthaler, Bernhard Waltl, Dominik Huth, Daniel Braun, Florian Matthes, Christoph Stocker, and Thomas Geiger. Extending thesauri using word embeddings and the intersection method. 06 2017.

[13] Michel Kana. *Representing text in natural language processing.* https://towardsdatascience.com/representing-text-in-natural-language-processing-1eead30e57d8, 2019.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* http://arxiv.org/abs/1810.04805, 2018.

[15] Rani Horev. *BERT Explained: State of the art language model for NLP.* https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270, 2018.

[16] Kaushal Trivedi. Introducing fastbert — a simple deep learning library for bert models, Sep 2019.

[17] Mohd Sanad Zaki Rizvi. *What is BERT: BERT For Text Classification.* https://www.analyticsvidhya.com/blog/2019/09/demystifying-bert-groundbreaking-nlp-framework/, 2019.

[18] Albert Au Yeung. *BERT - Tokenization and Encoding.* https://albertauyeung.github.io/2020/06/19/bert-tokenization.html, 2020.

[19] Edward Ma. *3 subword algorithms help to improve your NLP model performance.* https://medium.com/@makcedward/how-subword-helps-on-your-nlp-model-83dd1b836f46, 2019.

[20] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. *Transformers: State-of-the-Art Natural Language Processing.* https://www.aclweb.org/anthology/2020.emnlp-demos.6, 2020.

[21] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. CoRR*, abs/1910.10683, 2019.

[22] Rohan Jagtap. *T5: Text-To-Text Transfer Transformer.* https://towardsdatascience.com/t5-text-to-text-transfer-transformer-643f89e8905e, 2020.

[23] Synced. *Google T5 Explores the Limits of Transfer Learning.* https://medium.com/syncedreview/google-t5-explores-the-limits-of-transfer-learning-a87afbf2615b, 2019.

[24] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. *MASS: Masked Sequence to Sequence Pre-training for Language Generation. CoRR*, abs/1905.02450, 2019.

[25] Common Crawl. *Want to use our data? – Common Crawl.* `https://commoncrawl.org/the-data/`.

[26] Creative Commons Corporation. *List of Dirty, Naughty, Obscene, and Otherwise Bad Words.* `https://github.com/LDNOOBW/List-of-Dirty-Naughty-Obscene-and-Otherwise-Bad-Words`.

[27] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*, 2015. Software available from tensorflow.org.

[28] Dominic E. *Understand TensorFlow by mimicking its API from scratch.* `https://medium.com/@d3lm/understand-tensorflow-by-mimicking-its-api-from-scratch-faa55787170d`, 2019.

[29] Ray Johns. *PyTorch vs TensorFlow for your Python deep learning project.* `https://realpython.com/pytorch-vs-tensorflow/`, 2020.

[30] Sulabh Kunar. *Adapter Pattern.* `https://www.geeksforgeeks.org/adapter-pattern/`, 2018.

[31] Shashank Gupta. *Sentiment Analysis: Concept, Analysis and Applications.* `https://towardsdatascience.com/sentiment-analysis-concept-analysis-and-applications-6c94d6f58c17`, 2018.

[32] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. *Learning Word Vectors for Sentiment Analysis.* `http://www.aclweb.org/anthology/P11-1015`, 2011.

[33] Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. *Well-Read Students Learn Better: On the Importance of Pre-training Compact Models. arXiv preprint arXiv:1908.08962v2*, 2019.

[34] Jan Švec. *T5s - T5 made simple.* `https://github.com/honzas83/t5s`.

[35] Ivan Habernal, Tomáš Ptácek, and Josef Steinberger. *Sentiment analysis in czech social media using supervised machine learning.* In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 65–74, 2013.

[36] Jan Švec Jan Lehečka. *Czech BERT Models for Multi-label Document Classification.* 2021.

[37] Magda Ševčíková, Zdeněk Žabokrtský, and Oldřich Krůza. *Named Entities in Czech: Annotating Data and Developing NE Tagger.* In Václav Matoušek and Pavel Mautner, editors, *Lecture Notes in Artificial Intelligence, Proceedings of the 10th International Conference on Text, Speech and Dialogue*, volume 4629 of *Lecture Notes in Computer Science*, pages 188–195, Berlin / Heidelberg, 2007. Springer.

[38] F. Jurčíček, J. Zahradil, and L. Jelínek. *A human-human train timetable dialogue corpus. Interspeech Lisboa 2005*, pages 1525–1528, 2005.

[39] Marilyn Walker and Rebecca Passonneau. *DATE: A Dialogue Act Tagging Scheme for Evaluation of Spoken Dialogue Systems.* In *Proceedings of the First International Conference on Human Language Technology Research*, HLT '01, page 1–8, USA, 2001. Association for Computational Linguistics.

[40] Jan Švec. *Diskriminativní model pro porozumění mluvené řeči.* pages 99–107, 2013.

Souhrn dosažených výsledků

Dataset IMDB

BERT$_{BASE}$ model - IMDB						
Počet epoch	Batch size	Learning rate	F1	Přesnost	Úplnost	Délka sekvence
3	16	1.000e-05	0.8859	0.8794	0.8944	128
4	16	1.000e-05	0.8808	0.861	0.9084	128
2	16	1.000e-05	0.8773	0.8401	0.9329	128
4	32	1.000e-05	0.881	0.8922	0.8668	128
2	32	1.000e-05	0.8863	0.872	0.9056	128
3	32	1.000e-05	0.8787	0.9021	0.8499	128
2	64	1.000e-05	0.8816	0.8841	0.8782	128
3	64	1.000e-05	0.8831	0.8812	0.8855	128
4	64	1.000e-05	0.8766	0.8931	0.8558	128
3	16	2.000e-05	0.87	0.8275	0.9363	128
4	16	2.000e-05	0.877	0.9039	0.8436	128
2	16	2.000e-05	0.8805	0.863	0.9046	128
2	32	2.000e-05	0.8779	0.9079	0.8415	128
4	32	2.000e-05	0.8759	0.8961	0.8505	128
3	32	2.000e-05	0.8823	0.8795	0.886	128
4	64	2.000e-05	0.8813	0.8919	0.8679	128
2	64	2.000e-05	0.8864	0.8751	0.9015	128
3	64	2.000e-05	0.8849	0.8898	0.8788	128
3	16	3.000e-05	0.8621	0.9223	0.7922	128
2	16	3.000e-05	0.879	0.854	0.9148	128
4	16	3.000e-05	0.8712	0.913	0.8212	128
3	32	3.000e-05	0.8794	0.8908	0.865	128
2	32	3.000e-05	0.8874	0.8821	0.8944	128
4	32	3.000e-05	0.8808	0.871	0.894	128
4	64	3.000e-05	0.8802	0.8645	0.9018	128
3	64	3.000e-05	0.8822	0.8921	0.8695	128
2	64	3.000e-05	0.8718	0.8295	0.9372	128
4	16	4.000e-05	0.8653	0.9083	0.8135	128
3	16	4.000e-05	0.8661	0.8991	0.8253	128
2	16	4.000e-05	0.8761	0.8477	0.9175	128
4	32	4.000e-05	0.8609	0.9144	0.7976	128
2	32	4.000e-05	0.8751	0.8418	0.9245	128
3	32	4.000e-05	0.8722	0.8376	0.9242	128
3	64	4.000e-05	0.8774	0.8903	0.8609	128
2	64	4.000e-05	0.8844	0.8864	0.8818	128
4	64	4.000e-05	0.8863	0.827	0.9275	128
2	16	5.000e-05	0.8777	0.8859	0.8672	128
4	16	5.000e-05	0.8684	0.848	0.898	128
3	16	5.000e-05	0.8646	0.8649	0.864	128
3	32	5.000e-05	0.8798	0.8652	0.9001	128
4	32	5.000e-05	0.8696	0.889	0.8449	128
2	32	5.000e-05	0.8813	0.8513	0.9244	128
2	64	5.000e-05	0.8828	0.866	0.9057	128
4	64	5.000e-05	0.8744	0.8797	0.8673	128
3	64	5.000e-05	0.8796	0.8721	0.8896	128
3	16	1.000e-05	0.9354	0.9405	0.9297	512
4	16	1.000e-05	0.9365	0.9408	0.9315	512
2	16	1.000e-05	**0.9378**	0.9258	0.9519	512
4	32	1.000e-05	0.9345	0.9264	0.9441	512
2	32	1.000e-05	0.9347	0.9337	0.9357	512
3	32	1.000e-05	0.9357	0.9351	0.9364	512
2	64	1.000e-05	0.9339	0.9297	0.9388	512
3	64	1.000e-05	0.9345	0.9391	0.9293	512
4	64	1.000e-05	0.9351	0.9393	0.9305	512
3	16	2.000e-05	0.9332	0.9384	0.9272	512
4	16	2.000e-05	0.9348	0.9216	0.9505	512
2	16	2.000e-05	0.9365	0.936	0.937	512
2	32	2.000e-05	0.933	0.9532	0.9107	512
4	32	2.000e-05	0.9344	0.9315	0.9378	512
3	32	2.000e-05	0.9364	0.9373	0.9352	512
4	64	2.000e-05	0.9323	0.9406	0.9229	512
2	64	2.000e-05	0.9338	0.9209	0.9492	512

3	64	2.000e-05	0.9343	0.9375	0.9306	512
3	16	3.000e-05	0.9281	0.9509	0.9029	512
2	16	3.000e-05	0.9295	0.9198	0.9412	512
4	16	3.000e-05	0.9298	0.9068	0.9582	512
3	32	3.000e-05	0.9258	0.956	0.8928	512
2	32	3.000e-05	0.9347	0.9339	0.9356	512
4	32	3.000e-05	0.9357	0.9405	0.9302	512
4	64	3.000e-05	0.9331	0.9109	0.9602	512
3	64	3.000e-05	0.9352	0.9306	0.9405	512
2	64	3.000e-05	0.9355	0.9403	0.9301	512
4	16	4.000e-05	0.9255	0.9251	0.9261	512
3	16	4.000e-05	0.9295	0.9406	0.9169	512
2	16	4.000e-05	0.9299	0.9194	0.9425	512
4	32	4.000e-05	0.9325	0.9403	0.9236	512
2	32	4.000e-05	0.9349	0.9303	0.9404	512
3	32	4.000e-05	0.9349	0.9286	0.9423	512
3	64	4.000e-05	0.9283	0.9536	0.9005	512
2	64	4.000e-05	0.9315	0.9103	0.9574	512
4	64	4.000e-05	0.9358	0.9351	0.9366	512
2	16	5.000e-05	0.9247	0.8986	0.9577	512
4	16	5.000e-05	0.9265	0.939	0.9124	512
3	16	5.000e-05	0.9284	0.9363	0.9193	512
3	32	5.000e-05	0.9244	0.8933	0.9641	512
4	32	5.000e-05	0.9285	0.9412	0.9142	512
2	32	5.000e-05	0.933	0.9173	0.9519	512
2	64	5.000e-05	0.9288	0.9538	0.9014	512
4	64	5.000e-05	0.9301	0.9175	0.9451	512
3	64	5.000e-05	0.9325	0.935	0.9295	512

T5 model - IMDB						
Počet epoch	Steps per epoch	Learning rate	F1	Přesnost	Úplnost	Délka sekvence vstup, výstup
4	1400	1.000e-04	**0.9523**	0.9439	0.9617	3072,64
5	1200	1.000e-04	0.952	0.9516	0.9525	3072,64
6	1400	1.000e-04	0.952	0.9427	0.9625	3072,64
6	1200	1.000e-04	0.9517	0.9493	0.9544	3072,64
5	1200	5.000e-05	0.9515	0.949	0.9543	3072,64
7	1400	5.000e-05	0.9513	0.9473	0.9558	3072,64
8	1400	4.000e-05	0.9512	0.9467	0.9562	3072,64
4	1800	4.000e-05	0.9512	0.9533	0.9489	3072,64
8	1400	5.000e-05	0.9509	0.9533	0.9482	3072,64
7	1200	1.000e-04	0.9506	0.9421	0.9602	3072,64
2	1800	4.000e-05	0.9504	0.9547	0.9457	3072,64
4	1200	1.000e-04	0.95	0.9549	0.9445	3072,64
6	1200	5.000e-05	0.95	0.9494	0.9506	3072,64
5	1400	5.000e-05	0.9499	0.9351	0.967	3072,64
3	1400	5.000e-05	0.9498	0.9519	0.9475	3072,64
2	1400	1.000e-04	0.9496	0.9442	0.9557	3072,64
5	1800	4.000e-05	0.9492	0.9516	0.9466	3072,64
3	1400	1.000e-04	0.9491	0.9482	0.9502	3072,64
6	1200	3.000e-05	0.949	0.938	0.9617	3072,64
4	1400	5.000e-05	0.949	0.9543	0.9431	3072,64
7	1400	1.000e-04	0.9489	0.9627	0.9341	3072,64
3	1200	1.000e-04	0.9486	0.964	0.932	3072,64
4	1200	5.000e-05	0.9486	0.9624	0.9337	3072,64
3	1400	4.000e-05	0.9486	0.9376	0.9611	3072,64
7	1200	3.000e-05	0.9482	0.9316	0.9674	3072,64
7	1200	5.000e-05	0.9481	0.9316	0.9673	3072,64
5	1400	1.000e-04	0.9479	0.932	0.9664	3072,64
5	1200	3.000e-05	0.9478	0.9557	0.939	3072,64
4	1400	4.000e-05	0.9476	0.9375	0.9592	3072,64
2	1200	1.000e-04	0.9475	0.9477	0.9473	3072,64
6	1400	4.000e-05	0.9474	0.9296	0.9682	3072,64
2	1400	4.000e-05	0.9473	0.937	0.959	3072,64
4	1200	3.000e-05	0.9472	0.9616	0.9316	3072,64
7	1400	4.000e-05	0.9472	0.9354	0.9609	3072,64
8	1200	3.000e-05	0.9471	0.9278	0.9696	3072,64
3	1200	5.000e-05	0.9469	0.9607	0.9319	3072,64
3	1800	4.000e-05	0.9469	0.9312	0.9651	3072,64
2	1400	5.000e-05	0.9465	0.932	0.9633	3072,64
6	1400	5.000e-05	0.9458	0.9262	0.9688	3072,64
2	1200	5.000e-05	0.9443	0.964	0.9232	3072,64
3	1200	3.000e-05	0.9428	0.9651	0.919	3072,64
2	1200	3.000e-05	0.9426	0.9623	0.9213	3072,64
5	1400	4.000e-05	0.9412	0.9121	0.9767	3072,64

Dataset ČSFD

Český BERT model - ČSFD						
Počet epoch	Steps per epoch	Learning rate	F1	Přesnost	Úplnost	Délka sekvence
4	32	5.000e-05	0.9153	0.8929	0.9434	128
3	16	3.000e-05	0.9051	0.9312	0.8747	128
2	16	4.000e-05	0.9052	0.8868	0.9296	128
2	64	5.000e-05	0.9166	0.9426	0.8877	128
3	16	5.000e-05	0.9183	0.9298	0.9054	128
4	32	2.000e-05	0.9082	0.8847	0.9398	128
2	64	1.000e-05	0.9083	0.9089	0.9089	128
4	16	5.000e-05	0.9072	0.9119	0.9008	128
4	16	3.000e-05	0.9213	0.9462	0.8925	128
2	64	3.000e-05	0.9177	0.9097	0.9268	128
2	16	1.000e-05	0.9234	0.936	0.9096	128
3	32	2.000e-05	0.9173	0.8935	0.9476	128
4	32	3.000e-05	0.9193	0.9062	0.9355	128
4	16	4.000e-05	0.9203	0.9429	0.8956	128
4	16	1.000e-05	0.9162	0.9499	0.879	128
2	16	2.000e-05	0.9062	0.8838	0.9355	128
2	32	3.000e-05	0.9204	0.9406	0.918	128
4	64	2.000e-05	0.9281	0.952	0.9027	128
4	64	5.000e-05	0.9072	0.8858	0.9356	128
4	32	1.000e-05	0.9133	0.9131	0.9131	128
3	64	3.000e-05	0.9156	0.9069	0.9275	128
3	64	1.000e-05	0.9187	0.9179	0.9216	128
2	32	1.000e-05	0.9203	0.9315	0.9071	128
2	32	5.000e-05	0.9203	0.9082	0.9356	128
2	16	3.000e-05	0.9142	0.8872	0.9497	128
3	64	4.000e-05	0.9198	0.9146	0.9242	128
3	32	5.000e-05	0.9293	0.947	0.8958	128
3	32	3.000e-05	0.9234	0.9102	0.9395	128
3	64	2.000e-05	0.9093	0.8944	0.9296	128
3	16	4.000e-05	0.9183	0.9423	0.8909	128
4	32	4.000e-05	0.9192	0.8879	0.9596	128
4	64	3.000e-05	0.925	0.9305	0.9189	128
3	16	1.000e-05	0.9203	0.9315	0.9071	128
3	16	2.000e-05	0.9204	0.913	0.9296	128
4	64	1.000e-05	0.9177	0.9333	0.9004	128
2	64	2.000e-05	0.9167	0.9244	0.9091	128
3	32	1.000e-05	0.9274	0.9342	0.919	128
2	32	4.000e-05	0.9183	0.9368	0.8972	128
2	64	4.000e-05	0.9166	0.9393	0.8928	128
3	64	5.000e-05	0.9187	0.9138	0.9252	128
2	32	2.000e-05	0.9102	0.8866	0.9418	128
2	16	5.000e-05	0.9234	0.9048	0.9414	128
4	64	4.000e-05	0.924	0.9196	0.9292	128
4	16	2.000e-05	0.9031	0.8805	0.9336	128
3	32	4.000e-05	0.9103	0.9082	0.9137	128
4	32	5.000e-05	0.8939	0.8558	0.9475	512
3	16	3.000e-05	0.894	0.8659	0.9337	512
2	16	4.000e-05	0.9052	0.9244	0.8835	512
2	64	5.000e-05	0.9061	0.8664	0.9578	512
3	16	5.000e-05	0.9072	0.9257	0.8844	512
4	32	2.000e-05	0.9101	0.8794	0.8518	512
2	64	1.000e-05	0.9114	0.8964	0.9317	512
4	16	5.000e-05	0.9123	0.8956	0.9335	512
4	16	3.000e-05	0.9133	0.9085	0.9195	512
2	64	3.000e-05	0.9135	0.898	0.9335	512
2	16	1.000e-05	0.9143	0.9173	0.9118	512
3	32	2.000e-05	0.9163	0.9091	0.9256	512
4	32	3.000e-05	0.9173	0.9207	0.9133	512
4	16	4.000e-05	0.9173	0.9	0.9398	512
4	16	1.000e-05	0.9183	0.8969	0.9457	512
2	16	2.000e-05	0.9183	0.9315	0.9034	512
2	32	3.000e-05	0.9183	0.9045	0.9355	512
4	64	2.000e-05	0.9187	0.9036	0.9375	512
4	64	5.000e-05	0.9187	0.9249	0.9093	512
4	32	1.000e-05	0.9193	0.9317	0.9054	512
3	64	3.000e-05	0.9197	0.9427	0.8935	512
3	64	1.000e-05	0.9198	0.9255	0.9119	512
2	32	1.000e-05	0.9203	0.9371	0.9012	512
2	32	5.000e-05	0.9203	0.9098	0.9336	512
2	16	3.000e-05	0.9204	0.928	0.9111	512
3	64	4.000e-05	0.9208	0.9032	0.9442	512
3	32	5.000e-05	0.9212	0.9707	0.869	512
3	32	3.000e-05	0.9213	0.9467	0.8934	512
3	64	2.000e-05	0.9229	0.9371	0.9057	512
3	16	4.000e-05	0.9233	0.8975	0.9556	512
4	32	4.000e-05	0.9233	0.899	0.9535	512
4	64	3.000e-05	0.9239	0.9469	0.8973	512
3	16	1.000e-05	0.9243	0.9469	0.8992	512
3	16	2.000e-05	0.9244	0.9414	0.9054	512
4	64	1.000e-05	0.9249	0.9512	0.8956	512

2	64	2.000e-05	0.925	0.9384	0.9113	512
3	32	1.000e-05	0.9254	0.927	0.9232	512
2	32	4.000e-05	0.9254	0.945	0.903	512
2	64	4.000e-05	0.927	0.9518	0.9004	512
3	64	5.000e-05	0.9292	0.922	0.9374	512
2	32	2.000e-05	0.9294	0.9475	0.9093	512
2	16	5.000e-05	0.9294	0.9259	0.9333	512
4	64	4.000e-05	0.9302	0.9231	0.9367	512
4	16	2.000e-05	0.9314	0.9198	0.9457	512
3	32	4.000e-05	**0.9315**	0.9279	0.9354	512

Multilingual BERT model - ČSFD						
Počet epoch	Steps per epoch	Learning rate	F1	Přesnost	Úplnost	Délka sekvence
2	16	1.000e-05	0.9032	0.9097	0.8949	128
2	32	1.000e-05	0.8901	0.8712	0.9152	128
2	64	1.000e-05	0.881	0.847	0.9287	128
3	16	1.000e-05	0.8835	0.8399	0.9497	128
3	32	1.000e-05	0.8869	0.9231	0.8451	128
3	64	1.000e-05	0.8895	0.876	0.9087	128
4	16	1.000e-05	0.8866	0.8399	0.9555	128
4	32	1.000e-05	0.9032	0.9184	0.8851	128
4	64	1.000e-05	0.8895	0.8661	0.9205	128
2	16	2.000e-05	0.9083	0.8957	0.9229	128
2	32	2.000e-05	0.9011	0.8743	0.9376	128
2	64	2.000e-05	0.8979	0.8815	0.9184	128
3	16	2.000e-05	0.902	0.9462	0.8525	128
3	32	2.000e-05	0.9062	0.9006	0.9133	128
3	64	2.000e-05	0.8968	0.8743	0.9271	128
4	16	2.000e-05	0.8899	0.8548	0.9394	128
4	32	2.000e-05	0.9102	0.9145	0.9052	128
4	64	2.000e-05	0.9114	0.901	0.9253	128
2	16	3.000e-05	0.8669	0.8116	0.9576	128
2	32	3.000e-05	0.8991	0.8755	0.9317	128
2	64	3.000e-05	0.9007	0.9528	0.8434	128
3	16	3.000e-05	0.9112	0.8845	0.9453	128
3	32	3.000e-05	0.9083	0.9193	0.8952	128
3	64	3.000e-05	0.9083	0.9242	0.8896	128
4	16	3.000e-05	0.8951	0.8726	0.9254	128
4	32	3.000e-05	0.8898	0.8537	0.9415	128
4	64	3.000e-05	0.9042	0.8975	0.9125	128
2	16	4.000e-05	0.8731	0.9447	0.7939	128
2	32	4.000e-05	0.9042	0.9002	0.9093	128
2	64	4.000e-05	0.9038	0.867	0.9566	128
3	16	4.000e-05	0.892	0.8636	0.9315	128
3	32	4.000e-05	0.9062	0.9134	0.8968	128
3	64	4.000e-05	0.9073	0.9064	0.9083	128
4	16	4.000e-05	0.8752	0.8247	0.9557	128
4	32	4.000e-05	0.8918	0.8533	0.9477	128
4	64	4.000e-05	0.8968	0.9326	0.8574	128
2	16	5.000e-05	0.3324	0.498	1.0	128
2	32	5.000e-05	0.9113	0.8949	0.9312	128
2	64	5.000e-05	0.8934	0.9326	0.8466	128
3	16	5.000e-05	0.3338	0.501	1.0	128
3	32	5.000e-05	0.8921	0.8851	0.9012	128
3	64	5.000e-05	0.9019	0.8794	0.9339	128
4	16	5.000e-05	0.3329	0.499	1.0	128
4	32	5.000e-05	0.9113	0.9076	0.915	128
4	64	5.000e-05	0.9083	0.9043	0.9155	128
2	16	1.000e-05	0.9111	0.9492	0.8687	512
2	32	1.000e-05	0.9052	0.8821	0.9355	512
2	64	1.000e-05	0.9052	0.8902	0.9253	512
3	16	1.000e-05	0.9193	0.9428	0.8936	512
3	32	1.000e-05	0.9101	0.8806	0.9497	512
3	64	1.000e-05	0.9125	0.9044	0.9197	512
4	16	1.000e-05	0.9224	0.9082	0.9394	512
4	32	1.000e-05	0.9123	0.9345	0.8876	512
4	64	1.000e-05	0.9218	0.9383	0.9006	512
2	16	2.000e-05	0.875	0.8199	0.9637	512
2	32	2.000e-05	0.9103	0.898	0.9253	512
2	64	2.000e-05	0.9104	0.8944	0.9315	512
3	16	2.000e-05	0.9123	0.9004	0.9276	512
3	32	2.000e-05	0.9143	0.9037	0.9274	512
3	64	2.000e-05	0.9187	0.9304	0.903	512
4	16	2.000e-05	0.9234	0.9217	0.9254	512
4	32	2.000e-05	0.9244	0.922	0.9276	512
4	64	2.000e-05	0.9166	0.9038	0.9337	512
2	16	3.000e-05	0.9153	0.9363	0.8909	512
2	32	3.000e-05	0.9163	0.9349	0.8954	512
2	64	3.000e-05	0.9103	0.882	0.948	512
3	16	3.000e-05	0.9163	0.8942	0.9432	512
3	32	3.000e-05	0.8886	0.8477	0.9498	512

3	64	3.000e-05	0.9094	0.91	0.9081	512
4	16	3.000e-05	0.9103	0.898	0.9253	512
4	32	3.000e-05	0.9051	0.8743	0.9452	512
4	64	3.000e-05	0.903	0.9308	0.8706	512
2	16	4.000e-05	0.9032	0.91	0.8954	512
2	32	4.000e-05	0.904	0.869	0.9515	512
2	64	4.000e-05	0.902	0.881	0.9289	512
3	16	4.000e-05	0.8919	0.855	0.9433	512
3	32	4.000e-05	0.9163	0.9172	0.9153	512
3	64	4.000e-05	**0.9302**	0.9337	0.928	512
4	16	4.000e-05	0.8918	0.8527	0.9475	512
4	32	4.000e-05	0.9092	0.8876	0.9376	512
4	64	4.000e-05	0.9177	0.8992	0.9389	512
2	16	5.000e-05	0.8837	0.8485	0.9356	512
2	32	5.000e-05	0.869	0.8172	0.9537	512
2	64	5.000e-05	0.9156	0.9218	0.9083	512
3	16	5.000e-05	0.3329	0.499	1.0	512
3	32	5.000e-05	0.9103	0.8992	0.9229	512
3	64	5.000e-05	0.9125	0.9117	0.9155	512
4	16	5.000e-05	0.897	0.9357	0.8526	512
4	32	5.000e-05	0.897	0.94	0.8494	512
4	64	5.000e-05	0.9229	0.9139	0.9331	512

T5 model - ČSFD						
Počet epoch	Steps per epoch	Learning rate	F1	Přesnost	Úplnost	Délka sekvence vstup, výstup
2	1600	3.000e-04	0.938	0.938	0.938	3072,64
2	1200	5.000e-04	0.935	0.9466	0.922	3072,64
2	1400	5.000e-04	0.932	0.9219	0.944	3072,64
2	1600	5.000e-04	0.931	0.9168	0.948	3072,64
2	1000	1.000e-03	0.922	0.9154	0.93	3072,64
2	1000	5.000e-04	0.921	0.9072	0.938	3072,64
2	1600	1.000e-04	0.909	0.9252	0.89	3072,64
3	1400	5.000e-04	0.937	0.945	0.928	3072,64
3	1000	5.000e-04	0.933	0.927	0.94	3072,64
3	1200	5.000e-04	0.932	0.9186	0.948	3072,64
3	1600	3.000e-04	0.93	0.9249	0.936	3072,64
3	1600	5.000e-04	0.928	0.9115	0.948	3072,64
3	1000	1.000e-03	0.926	0.9419	0.908	3072,64
3	1000	1.000e-04	0.901	0.8986	0.904	3072,64
4	1200	5.000e-04	0.941	0.9455	0.936	3072,64
4	1600	3.000e-04	0.937	0.9293	0.946	3072,64
4	1000	5.000e-04	0.937	0.9293	0.946	3072,64
4	1600	5.000e-04	0.934	0.9411	0.926	3072,64
4	1400	5.000e-04	0.9289	0.898	0.968	3072,64
4	1000	1.000e-03	0.925	0.9242	0.926	3072,64
5	1000	5.000e-04	0.943	0.9421	0.944	3072,64
5	1400	5.000e-04	0.942	0.9368	0.948	3072,64
5	1600	5.000e-04	0.94	0.949	0.93	3072,64
5	1600	3.000e-04	0.938	0.9228	0.956	3072,64
5	1000	1.000e-03	0.932	0.9303	0.934	3072,64
5	1200	5.000e-04	0.9319	0.906	0.964	3072,64
6	1400	5.000e-04	**0.947**	0.9552	0.938	3072,64
6	1600	5.000e-04	0.946	0.9425	0.95	3072,64
6	1200	5.000e-04	0.942	0.93	0.956	3072,64
6	1000	5.000e-04	0.939	0.9507	0.926	3072,64
6	1600	1.000e-04	0.934	0.9272	0.942	3072,64
6	1000	1.000e-03	0.928	0.918	0.94	3072,64
6	1000	1.000e-04	0.925	0.9242	0.926	3072,64
7	1000	5.000e-04	0.942	0.942	0.942	3072,64
7	1400	5.000e-04	0.939	0.9213	0.96	3072,64
7	1200	5.000e-04	0.938	0.9415	0.934	3072,64
7	1600	5.000e-04	0.934	0.934	0.934	3072,64
7	1000	1.000e-03	0.8974	0.8455	0.974	3072,64
8	1600	3.000e-04	0.946	0.9407	0.952	3072,64
8	1600	5.000e-04	0.941	0.9282	0.956	3072,64
9	1600	1.000e-04	0.946	0.9588	0.932	3072,64
9	1400	1.000e-04	0.936	0.9241	0.95	3072,64
9	1000	1.000e-04	0.925	0.9159	0.936	3072,64
12	1600	1.000e-04	0.945	0.932	0.96	3072,64
12	1600	3.000e-04	0.944	0.9302	0.96	3072,64
12	1600	3.000e-04	0.944	0.9405	0.948	3072,64

Dataset CNEC

Český BERT model - CNEC 7 entit						
Počet epoch	Steps per epoch	Learning rate	F1	Přesnost	Úplnost	Délka sekvence
10	32	1.000e-05	0.7794	0.7636	0.796	128
14	32	1.000e-05	0.791	0.7783	0.8041	128
10	32	2.000e-05	0.7948	0.784	0.806	128
10	16	1.000e-05	0.7957	0.7822	0.8097	128
12	32	1.000e-05	0.7969	0.7804	0.8141	128
16	32	1.000e-05	0.7997	0.7861	0.8137	128
16	16	1.000e-05	0.8001	0.7898	0.8108	128
18	32	1.000e-05	0.8049	0.793	0.8171	128
12	16	1.000e-05	0.8059	0.7929	0.8193	128
32	32	1.000e-05	0.8061	0.7982	0.8141	128
22	32	1.000e-05	0.8073	0.7953	0.8197	128
18	16	1.000e-05	0.8075	0.7991	0.816	128
12	16	2.000e-05	0.8109	0.8027	0.8193	128
14	16	1.000e-05	0.8114	0.8012	0.8219	128
10	32	3.000e-05	0.8129	0.8031	0.823	128
12	32	3.000e-05	0.8132	0.8051	0.8215	128
14	32	2.000e-05	0.8133	0.8059	0.8208	128
22	16	1.000e-05	0.8136	0.8033	0.8241	128
16	32	2.000e-05	0.8146	0.8068	0.8226	128
18	32	2.000e-05	0.8172	0.8077	0.8271	128
32	16	1.000e-05	0.8182	0.8114	0.8252	128
12	32	2.000e-05	0.8183	0.8065	0.8304	128
28	16	1.000e-05	0.8186	0.8101	0.8274	128
16	16	2.000e-05	0.8192	0.8122	0.8263	128
28	32	1.000e-05	0.8194	0.8098	0.8293	128
18	32	4.000e-05	0.8214	0.8176	0.8252	128
10	16	2.000e-05	0.8221	0.8153	0.8289	128
14	16	2.000e-05	0.8224	0.8159	0.8289	128
14	32	4.000e-05	0.8226	0.8143	0.8311	128
16	32	4.000e-05	0.8228	0.8168	0.8289	128
10	16	3.000e-05	0.8228	0.8168	0.8289	128
28	32	4.000e-05	0.823	0.8119	0.8344	128
22	16	2.000e-05	0.8236	0.8158	0.8315	128
22	32	2.000e-05	0.8238	0.8174	0.8304	128
14	16	4.000e-05	0.8238	0.8166	0.8311	128
10	32	5.000e-05	0.8244	0.8171	0.8319	128
10	32	4.000e-05	0.8245	0.8193	0.8296	128
14	16	3.000e-05	0.8245	0.8176	0.8315	128
14	32	3.000e-05	0.8246	0.8156	0.8337	128
22	32	4.000e-05	0.8246	0.8161	0.8333	128
16	32	3.000e-05	0.8247	0.8167	0.833	128
12	32	5.000e-05	0.8249	0.8145	0.8356	128
16	16	3.000e-05	0.825	0.8182	0.8319	128
10	16	4.000e-05	0.8253	0.8188	0.8319	128
28	32	3.000e-05	0.8254	0.8184	0.8326	128
18	16	2.000e-05	0.8258	0.8197	0.8319	128
18	32	5.000e-05	0.8259	0.8168	0.8352	128
32	16	2.000e-05	0.8261	0.8176	0.8348	128
18	16	3.000e-05	0.8267	0.8173	0.8363	128
32	32	3.000e-05	0.8269	0.8206	0.8333	128
32	32	4.000e-05	0.8271	0.8189	0.8356	128
12	16	5.000e-05	0.8273	0.8192	0.8356	128
32	32	2.000e-05	0.8278	0.8201	0.8356	128
12	32	4.000e-05	0.8282	0.8235	0.833	128
12	16	3.000e-05	0.8284	0.8236	0.8333	128
28	32	2.000e-05	0.8286	0.8214	0.8359	128
22	32	3.000e-05	0.8289	0.8192	0.8389	128
12	16	4.000e-05	0.8289	0.8235	0.8344	128
32	16	5.000e-05	0.8291	0.8228	0.8356	128
22	16	5.000e-05	0.8292	0.8208	0.8378	128
22	32	5.000e-05	0.8301	0.8218	0.8385	128
16	16	4.000e-05	0.8302	0.8231	0.8374	128
10	16	5.000e-05	0.8309	0.8264	0.8356	128
16	32	5.000e-05	0.8312	0.8211	0.8415	128
14	32	5.000e-05	0.8325	0.8276	0.8374	128
16	16	5.000e-05	0.8328	0.8262	0.8396	128
28	16	4.000e-05	0.833	0.8291	0.837	128
32	16	4.000e-05	0.8334	0.8276	0.8392	128
18	32	3.000e-05	0.8339	0.8273	0.8407	128
28	32	5.000e-05	0.8342	0.8253	0.8433	128
28	16	5.000e-05	0.8346	0.829	0.8404	128
32	16	3.000e-05	0.8352	0.8261	0.8444	128
28	16	2.000e-05	0.8355	0.8282	0.8429	128
14	16	5.000e-05	0.8355	0.8257	0.8455	128
22	16	4.000e-05	0.8358	0.8309	0.8407	128
22	16	3.000e-05	0.8359	0.8294	0.8426	128
18	16	4.000e-05	0.8359	0.8312	0.8407	128
28	16	3.000e-05	0.8361	0.8297	0.8426	128
18	16	5.000e-05	0.8361	0.8287	0.8437	128
32	32	5.000e-05	**0.8363**	0.8308	0.8418	128

Multilingual BERT model - CNEC 7 entit						
Počet epoch	Steps per epoch	Learning rate	F1	Přesnost	Úplnost	Délka sekvence
32	16	4.000e-05	**0.8184**	0.8165	0.8204	128
18	32	3.000e-05	0.8182	0.8126	0.8252	128
12	16	3.000e-05	0.8172	0.8133	0.8211	128
16	16	2.000e-05	0.8166	0.8104	0.823	128
28	32	3.000e-05	0.8161	0.8083	0.8241	128
32	32	3.000e-05	0.8161	0.8105	0.8219	128
12	32	4.000e-05	0.816	0.8117	0.8204	128
28	16	5.000e-05	0.8149	0.8114	0.8186	128
18	32	4.000e-05	0.8143	0.8119	0.8167	128
28	16	2.000e-05	0.8141	0.8079	0.8204	128
14	16	2.000e-05	0.814	0.8098	0.8182	128
16	32	5.000e-05	0.8137	0.8036	0.8241	128
22	32	3.000e-05	0.8136	0.8079	0.8193	128
28	32	4.000e-05	0.8133	0.8106	0.816	128
18	16	1.000e-05	0.8127	0.8069	0.8186	128
32	16	3.000e-05	0.8126	0.8112	0.8211	128
14	32	3.000e-05	0.8125	0.8076	0.8174	128
10	32	5.000e-05	0.8123	0.8068	0.8178	128
28	16	4.000e-05	0.8123	0.8068	0.8178	128
32	16	2.000e-05	0.8122	0.8023	0.8126	128
14	16	3.000e-05	0.8122	0.8095	0.8149	128
32	32	2.000e-05	0.812	0.8052	0.8189	128
28	32	5.000e-05	0.8117	0.8056	0.8178	128
12	16	2.000e-05	0.8117	0.8054	0.8182	128
28	16	3.000e-05	0.8117	0.8056	0.8178	128
22	32	1.000e-05	0.8113	0.7999	0.823	128
22	16	4.000e-05	0.8113	0.8099	0.8126	128
32	32	5.000e-05	0.8109	0.8088	0.813	128
10	16	4.000e-05	0.8104	0.8074	0.8134	128
14	32	4.000e-05	0.8103	0.8047	0.816	128
12	16	4.000e-05	0.8103	0.8094	0.8112	128
16	16	5.000e-05	0.8103	0.8047	0.816	128
32	32	1.000e-05	0.8102	0.8017	0.8189	128
14	16	5.000e-05	0.8099	0.8036	0.8163	128
28	32	1.000e-05	0.8096	0.8037	0.8156	128
16	16	3.000e-05	0.8096	0.8067	0.8126	128
18	16	3.000e-05	0.8095	0.8034	0.8156	128
28	16	1.000e-05	0.8094	0.8036	0.8152	128
22	16	2.000e-05	0.8094	0.8051	0.8137	128
22	16	3.000e-05	0.8094	0.8022	0.8167	128
14	32	2.000e-05	0.8093	0.8	0.8189	128
28	32	2.000e-05	0.8093	0.8017	0.8171	128
22	16	1.000e-05	0.8093	0.8017	0.8171	128
10	32	2.000e-05	0.8092	0.8026	0.816	128
18	32	2.000e-05	0.8091	0.8038	0.8145	128
12	16	5.000e-05	0.8091	0.8031	0.8152	128
14	32	5.000e-05	0.8089	0.8006	0.8174	128
18	32	1.000e-05	0.8087	0.7976	0.82	128
10	16	2.000e-05	0.8087	0.8051	0.8123	128
12	32	2.000e-05	0.8081	0.8001	0.8163	128
18	16	4.000e-05	0.8078	0.8012	0.8145	128
16	32	4.000e-05	0.8077	0.8051	0.8104	128
32	16	1.000e-05	0.8074	0.8023	0.8126	128
22	32	4.000e-05	0.8072	0.8029	0.8115	128
16	16	1.000e-05	0.8072	0.7991	0.8156	128
10	16	3.000e-05	0.8072	0.7998	0.8149	128
10	32	4.000e-05	0.8068	0.7989	0.8149	128
32	32	4.000e-05	0.8057	0.8017	0.8097	128
14	16	1.000e-05	0.8057	0.8	0.8115	128
10	16	5.000e-05	0.8056	0.8004	0.8108	128
18	16	5.000e-05	0.8056	0.7995	0.8119	128
22	32	2.000e-05	0.8055	0.8007	0.8104	128
22	16	5.000e-05	0.8055	0.8038	0.8071	128
18	32	5.000e-05	0.8054	0.8004	0.8104	128
16	32	3.000e-05	0.805	0.7969	0.8134	128
12	32	5.000e-05	0.8048	0.7977	0.8119	128
16	32	2.000e-05	0.8046	0.7982	0.8112	128
12	32	3.000e-05	0.8045	0.7984	0.8108	128
22	32	5.000e-05	0.8033	0.7956	0.8112	128
14	16	4.000e-05	0.8032	0.7982	0.8082	128
12	16	1.000e-05	0.8029	0.7954	0.8104	128
18	16	2.000e-05	0.8028	0.8007	0.8049	128
16	16	4.000e-05	0.8024	0.7984	0.8064	128
10	32	3.000e-05	0.8021	0.7978	0.8064	128
10	16	1.000e-05	0.802	0.7935	0.8108	128
32	16	5.000e-05	0.8014	0.7969	0.806	128
10	32	1.000e-05	0.8012	0.7918	0.8108	128
12	32	1.000e-05	0.8004	0.7911	0.8101	128
16	32	1.000e-05	0.8004	0.7934	0.8075	128
14	32	1.000e-05	0.7977	0.7892	0.8064	128

T5 model - CNEC 42 entit						
Počet epoch	Steps per epoch	Learning rate	F1	Přesnost	Úplnost	Délka sekvence vstup, výstup
4	1200	1.000e-03	0.6892	0.6925	0.6859	3072,128
5	1200	1.000e-03	0.7101	0.6931	0.7279	3072,128
6	1200	1.000e-03	0.7575	0.7647	0.7505	3072,128
7	1200	1.000e-03	0.7503	0.7589	0.7418	3072,128
8	1200	1.000e-03	0.7236	0.7174	0.7299	3072,128
9	1200	1.000e-03	0.7304	0.7329	0.7279	3072,128
15	1200	1.000e-03	0.6829	0.6259	0.7513	3072,128
4	1700	1.000e-03	0.6664	0.6306	0.7065	3072,128
5	1700	1.000e-03	0.7282	0.7242	0.7323	3072,128
6	1700	1.000e-03	0.6963	0.7117	0.6816	3072,128
7	1700	1.000e-03	0.7367	0.7283	0.7453	3072,128
8	1700	1.000e-03	0.7065	0.6668	0.7513	3072,128
9	1700	1.000e-03	0.7156	0.6919	0.741	3072,128
15	1700	1.000e-03	0.7023	0.6818	0.724	3072,128
4	700	1.000e-03	0.6794	0.7033	0.657	3072,128
5	700	1.000e-03	0.6935	0.7328	0.6582	3072,128
6	700	1.000e-03	0.6822	0.7112	0.6554	3072,128
7	700	1.000e-03	0.723	0.7169	0.7291	3072,128
8	700	1.000e-03	0.71	0.6686	0.7568	3072,128
9	700	1.000e-03	0.7136	0.6818	0.7485	3072,128
15	700	1.000e-03	0.7407	0.7242	0.758	3072,128
4	1400	1.000e-03	0.7239	0.7219	0.7259	3072,128
5	1400	1.000e-03	0.7478	0.7635	0.7327	3072,128
6	1400	1.000e-03	0.733	0.7337	0.7323	3072,128
7	1400	1.000e-03	0.7272	0.6992	0.7576	3072,128
8	1400	1.000e-03	0.6788	0.6188	0.7517	3072,128
9	1400	1.000e-03	0.7486	0.7593	0.7382	3072,128
15	1400	1.000e-03	0.717	0.673	0.7671	3072,128
4	1200	5.000e-04	0.7465	0.7414	0.7517	3072,128
5	1200	5.000e-04	0.7383	0.72	0.7576	3072,128
6	1200	5.000e-04	0.7696	0.7774	0.762	3072,128
7	1200	5.000e-04	0.7487	0.7435	0.7541	3072,128
8	1200	5.000e-04	0.7601	0.7591	0.7612	3072,128
9	1200	5.000e-04	0.7735	0.7904	0.7572	3072,128
15	1200	5.000e-04	0.7678	0.7649	0.7708	3072,128
4	1700	5.000e-04	0.7209	0.7446	0.6986	3072,128
5	1700	5.000e-04	0.7648	0.7822	0.7481	3072,128
6	1700	5.000e-04	0.716	0.7191	0.7129	3072,128
7	1700	5.000e-04	0.7608	0.7665	0.7552	3072,128
8	1700	5.000e-04	0.7539	0.7631	0.745	3072,128
9	1700	5.000e-04	0.7435	0.7245	0.7636	3072,128
15	1700	5.000e-04	0.7847	0.8108	0.7604	3072,128
20	1700	5.000e-04	0.7671	0.7671	0.7671	3072,128
4	700	5.000e-04	0.7022	0.741	0.6673	3072,128
5	700	5.000e-04	0.7042	0.751	0.663	3072,128
6	700	5.000e-04	0.7086	0.7177	0.6998	3072,128
7	700	5.000e-04	0.7213	0.7199	0.7228	3072,128
8	700	5.000e-04	0.7525	0.7578	0.7473	3072,128
9	700	5.000e-04	0.757	0.7691	0.7453	3072,128
15	700	5.000e-04	0.7772	0.7947	0.7604	3072,128
4	1400	5.000e-04	0.7638	0.7776	0.7505	3072,128
5	1400	5.000e-04	0.7561	0.7511	0.7612	3072,128
6	1400	5.000e-04	**0.7871**	0.797	0.7774	3072,128
7	1400	5.000e-04	0.7711	0.7792	0.7632	3072,128
8	1400	5.000e-04	0.7583	0.7555	0.7612	3072,128
9	1400	5.000e-04	0.7663	0.7711	0.716	3072,128
15	1400	5.000e-04	0.7631	0.7576	0.7687	3072,128

Český BERT model - CNEC 42 entit						
Počet epoch	Steps per epoch	Learning rate	F1	Přesnost	Úplnost	Délka sekvence
10	32	3.000e-05	0.8024	0.7889	0.8164	128
12	32	3.000e-05	0.8005	0.784	0.8176	128
14	32	3.000e-05	0.8038	0.7894	0.8188	128
16	32	3.000e-05	0.8057	0.7912	0.8208	128
18	32	3.000e-05	0.8152	0.8063	0.8244	128
22	32	3.000e-05	0.8174	0.8078	0.8273	128
28	32	3.000e-05	0.8156	0.8058	0.8256	128
32	32	3.000e-05	0.8202	0.8159	0.8245	128
10	32	4.000e-05	0.7959	0.7776	0.8132	128
12	32	4.000e-05	0.8066	0.7915	0.8223	128
14	32	4.000e-05	0.8086	0.7936	0.8241	128
16	32	4.000e-05	0.8087	0.7963	0.8214	128

18	32	4.000e-05	0.8178	0.8071	0.8288	128
22	32	4.000e-05	0.8145	0.8043	0.825	128
28	32	4.000e-05	0.8176	0.8084	0.827	128
32	32	4.000e-05	0.8166	0.8093	0.8241	128
10	32	2.000e-05	0.7822	0.7643	0.8011	128
12	32	2.000e-05	0.7954	0.7795	0.812	128
14	32	2.000e-05	0.7964	0.7806	0.8129	128
16	32	2.000e-05	0.8002	0.7859	0.8149	128
18	32	2.000e-05	0.8057	0.7906	0.8214	128
22	32	2.000e-05	0.8104	0.7993	0.8217	128
28	32	2.000e-05	0.8075	0.7999	0.8152	128
32	32	2.000e-05	**0.8246**	0.8134	0.8362	128
10	32	1.000e-05	0.7269	0.7168	0.7373	128
12	32	1.000e-05	0.7413	0.7294	0.7535	128
14	32	1.000e-05	0.746	0.7252	0.768	128
16	32	1.000e-05	0.7749	0.7562	0.7946	128
18	32	1.000e-05	0.7898	0.7704	0.8102	128
22	32	1.000e-05	0.7953	0.7835	0.8076	128
28	32	1.000e-05	0.7966	0.7805	0.8135	128
32	32	1.000e-05	0.8072	0.795	0.8197	128
10	32	5.000e-05	0.7971	0.7814	0.8135	128
12	32	5.000e-05	0.8027	0.787	0.8191	128
14	32	5.000e-05	0.8098	0.7963	0.8238	128
16	32	5.000e-05	0.8104	0.7993	0.8217	128
18	32	5.000e-05	0.8111	0.8017	0.8208	128
22	32	5.000e-05	0.8205	0.8127	0.8285	128
28	32	5.000e-05	0.8139	0.8058	0.822	128
32	32	5.000e-05	0.8222	0.8154	0.8291	128
10	16	3.000e-05	0.8052	0.7912	0.8197	128
12	16	3.000e-05	0.8055	0.7867	0.8253	128
14	16	3.000e-05	0.8035	0.7842	0.8238	128
16	16	3.000e-05	0.8096	0.7945	0.8253	128
18	16	3.000e-05	0.816	0.8006	0.8321	128
22	16	3.000e-05	0.814	0.8041	0.8241	128
28	16	3.000e-05	0.8132	0.8043	0.8223	128
32	16	3.000e-05	0.8221	0.8138	0.8306	128
10	16	4.000e-05	0.8091	0.7931	0.8259	128
12	16	4.000e-05	0.8105	0.7947	0.827	128
14	16	4.000e-05	0.8027	0.7881	0.8179	128
16	16	4.000e-05	0.8097	0.7948	0.8253	128
18	16	4.000e-05	0.809	0.7958	0.8226	128
22	16	4.000e-05	0.8187	0.8098	0.8279	128
28	16	4.000e-05	0.8198	0.8125	0.8273	128
32	16	4.000e-05	0.8226	0.8141	0.8312	128
10	16	2.000e-05	0.8015	0.7895	0.8138	128
12	16	2.000e-05	0.7891	0.7753	0.8034	128
14	16	2.000e-05	0.8105	0.7958	0.8259	128
16	16	2.000e-05	0.809	0.7929	0.8259	128
18	16	2.000e-05	0.8032	0.7835	0.8238	128
22	16	2.000e-05	0.8104	0.7986	0.8226	128
28	16	2.000e-05	0.8188	0.809	0.8288	128
32	16	2.000e-05	0.8236	0.8164	0.8309	128
10	16	1.000e-05	0.7723	0.7528	0.7928	128
12	16	1.000e-05	0.7798	0.761	0.7996	128
14	16	1.000e-05	0.7892	0.7714	0.8079	128
16	16	1.000e-05	0.8003	0.7856	0.8155	128
18	16	1.000e-05	0.7942	0.7766	0.8126	128
22	16	1.000e-05	0.7992	0.7852	0.8138	128
28	16	1.000e-05	0.8098	0.7974	0.8226	128
32	16	1.000e-05	0.8137	0.7997	0.8282	128
10	16	5.000e-05	0.8074	0.7897	0.8259	128
12	16	5.000e-05	0.8088	0.7952	0.8229	128
14	16	5.000e-05	0.815	0.8003	0.8303	128
16	16	5.000e-05	0.8053	0.7925	0.8185	128
18	16	5.000e-05	0.8191	0.8087	0.8297	128
22	16	5.000e-05	0.8173	0.8121	0.8226	128
28	16	5.000e-05	0.8221	0.8171	0.827	128
32	16	5.000e-05	0.8168	0.8117	0.822	128

Multilingual BERT model - CNEC 42 entit						
Počet epoch	Steps per epoch	Learning rate	F1	Přesnost	Úplnost	Délka sekvence
10	32	3.000e-05	0.7914	0.7883	0.7946	128
12	32	3.000e-05	0.8137	0.8087	0.8188	128
14	32	3.000e-05	0.8104	0.8077	0.8132	128
16	32	3.000e-05	0.8152	0.811	0.8194	128
18	32	3.000e-05	0.8114	0.8114	0.8114	128
22	32	3.000e-05	0.8206	0.8239	0.8173	128
28	32	3.000e-05	0.8191	0.8195	0.8188	128
32	32	3.000e-05	0.82	0.8218	0.8182	128
10	32	4.000e-05	0.8047	0.7988	0.8108	128
12	32	4.000e-05	0.8096	0.8064	0.8129	128

14	32	4.000e-05	0.8026	0.7965	0.8087	128
16	32	4.000e-05	0.8079	0.8047	0.8111	128
18	32	4.000e-05	0.8095	0.813	0.8061	128
22	32	4.000e-05	0.8168	0.8198	0.8138	128
28	32	4.000e-05	**0.8225**	0.8278	0.8173	128
32	32	4.000e-05	0.8126	0.8163	0.809	128
10	32	2.000e-05	0.7855	0.7758	0.7955	128
12	32	2.000e-05	0.7975	0.7936	0.8014	128
14	32	2.000e-05	0.7979	0.7919	0.804	128
16	32	2.000e-05	0.8123	0.8067	0.8179	128
18	32	2.000e-05	0.8124	0.8105	0.8143	128
22	32	2.000e-05	0.8073	0.8079	0.8067	128
28	32	2.000e-05	0.8183	0.8178	0.8188	128
32	32	2.000e-05	0.8197	0.8195	0.82	128
10	32	1.000e-05	0.7541	0.7511	0.7571	128
12	32	1.000e-05	0.7763	0.7726	0.7801	128
14	32	1.000e-05	0.7815	0.7736	0.7896	128
16	32	1.000e-05	0.7924	0.7879	0.7969	128
18	32	1.000e-05	0.798	0.7938	0.8022	128
22	32	1.000e-05	0.8037	0.8002	0.8073	128
28	32	1.000e-05	0.8121	0.8104	0.8138	128
32	32	1.000e-05	0.8138	0.8117	0.8158	128
10	32	5.000e-05	0.798	0.7955	0.8005	128
12	32	5.000e-05	0.8057	0.7999	0.8117	128
14	32	5.000e-05	0.8078	0.8031	0.8126	128
16	32	5.000e-05	0.8158	0.8166	0.8149	128
18	32	5.000e-05	0.8178	0.8197	0.8158	128
22	32	5.000e-05	0.81	0.814	0.8061	128
28	32	5.000e-05	0.8131	0.8158	0.8105	128
32	32	5.000e-05	0.8063	0.809	0.8037	128
10	16	3.000e-05	0.8142	0.8109	0.8176	128
12	16	3.000e-05	0.806	0.8003	0.8117	128
14	16	3.000e-05	0.8113	0.8054	0.8173	128
16	16	3.000e-05	0.8113	0.8062	0.8164	128
18	16	3.000e-05	0.8083	0.805	0.8117	128
22	16	3.000e-05	0.8185	0.8239	0.8132	128
28	16	3.000e-05	0.8141	0.8159	0.8123	128
32	16	3.000e-05	0.8107	0.8126	0.8087	128
10	16	4.000e-05	0.8042	0.798	0.8105	128
12	16	4.000e-05	0.8087	0.802	0.8155	128
14	16	4.000e-05	0.8077	0.8021	0.8135	128
16	16	4.000e-05	0.8064	0.7997	0.8132	128
18	16	4.000e-05	0.8063	0.8006	0.812	128
22	16	4.000e-05	0.8121	0.8141	0.8102	128
28	16	4.000e-05	0.8028	0.8064	0.7993	128
32	16	4.000e-05	0.8143	0.8133	0.8152	128
10	16	2.000e-05	0.8122	0.8069	0.8176	128
12	16	2.000e-05	0.8135	0.8077	0.8194	128
14	16	2.000e-05	0.8095	0.8033	0.8158	128
16	16	2.000e-05	0.8142	0.8097	0.8188	128
18	16	2.000e-05	0.8153	0.8067	0.8241	128
22	16	2.000e-05	0.8195	0.8182	0.8208	128
28	16	2.000e-05	0.8086	0.8087	0.8084	128
32	16	2.000e-05	0.8207	0.82	0.8214	128
10	16	1.000e-05	0.7893	0.7849	0.7937	128
12	16	1.000e-05	0.7954	0.7923	0.7984	128
14	16	1.000e-05	0.7962	0.7867	0.8058	128
16	16	1.000e-05	0.8038	0.8	0.8076	128
18	16	1.000e-05	0.804	0.7976	0.8105	128
22	16	1.000e-05	0.8103	0.8067	0.814	128
28	16	1.000e-05	0.8112	0.8077	0.8146	128
32	16	1.000e-05	0.8158	0.8132	0.8185	128
10	16	5.000e-05	0.8038	0.7981	0.8096	128
12	16	5.000e-05	0.8055	0.7994	0.8117	128
14	16	5.000e-05	0.8033	0.7977	0.809	128
16	16	5.000e-05	0.8049	0.8019	0.8079	128
18	16	5.000e-05	0.8089	0.8058	0.812	128
22	16	5.000e-05	0.8041	0.805	0.8031	128
28	16	5.000e-05	0.8033	0.8066	0.7981	128
32	16	5.000e-05	0.81	0.8159	0.8043	128

T5 model - CNEC 42 entit						
Počet epoch	Steps per epoch	Learning rate	F1	Přesnost	Úplnost	Délka sekvence vstup, výstup
6	1600	1.000e-03	0.6973	0.7329	0.665	3072,128
10	2000	1.000e-04	0.6981	0.7761	0.6343	3072,128
7	2000	5.000e-04	0.7018	0.7632	0.6496	3072,128
20	1200	5.000e-04	0.7022	0.7757	0.6414	3072,128
10	1600	5.000e-04	0.7033	0.768	0.6487	3072,128
9	2000	5.000e-04	0.7033	0.7524	0.6602	3072,128
5	1600	1.000e-03	0.7035	0.7373	0.6727	3072,128

10	1400	5.000e-04	0.7037	0.7529	0.6606	3072,128
70	100	5.000e-04	0.7045	0.7662	0.6652	3072,128
15	1600	5.000e-04	0.7056	0.7602	0.6583	3072,128
70	100	1.000e-03	0.7063	0.7709	0.6517	3072,128
4	2000	5.000e-04	0.7064	0.746	0.6709	3072,128
20	1600	5.000e-04	0.7086	0.7564	0.6666	3072,128
6	1600	5.000e-04	0.7088	0.7415	0.6789	3072,128
5	2000	5.000e-04	0.7094	0.7486	0.6741	3072,128
4	1600	5.000e-04	0.712	0.7461	0.6809	3072,128
15	2000	1.000e-04	0.712	0.7686	0.6632	3072,128
25	1600	5.000e-04	0.7126	0.7612	0.6698	3072,128
15	1400	5.000e-04	0.7132	0.7642	0.6686	3072,128
30	1200	5.000e-04	0.7133	0.7611	0.6677	3072,128
9	2000	3.000e-04	0.7133	0.771	0.6636	3072,128
10	2000	3.000e-04	0.7134	0.7768	0.6595	3072,128
30	2000	1.000e-04	0.7142	0.7784	0.6598	3072,128
30	1600	5.000e-04	0.7164	0.7684	0.6709	3072,128
30	2100	3.000e-04	0.719	0.7781	0.6682	3072,128
10	2100	3.000e-04	0.7193	0.7766	0.6698	3072,128
25	2000	1.000e-04	0.7195	0.785	0.6641	3072,128
15	2100	3.000e-04	0.7201	0.7796	0.669	3072,128
25	2100	3.000e-04	0.7203	0.7806	0.6687	3072,128
30	1400	5.000e-04	0.721	0.7784	0.6715	3072,128
20	2100	3.000e-04	0.7215	0.7836	0.6685	3072,128
5	1600	5.000e-04	0.7227	0.7639	0.6857	3072,128
20	2000	1.000e-04	0.7232	0.7906	0.6663	3072,128
25	1400	5.000e-04	0.7235	0.7857	0.6705	3072,128
15	2000	3.000e-04	0.7242	0.7825	0.674	3072,128
30	2000	3.000e-04	0.7246	0.7767	0.679	3072,128
20	1400	5.000e-04	0.7254	0.7905	0.6701	3072,128
6	2000	5.000e-04	0.726	0.7638	0.6919	3072,128
25	2000	3.000e-04	0.7271	0.7832	0.6785	3072,128
20	2000	3.000e-04	0.7305	0.7905	0.679	3072,128
10	2400	1.000e-03	0.7332	0.7925	0.6821	3072,128
40	500	5.000e-04	0.7335	0.7675	0.7025	3072,128
30	2400	5.000e-04	0.7345	0.7821	0.6923	3072,128
10	2400	5.000e-04	0.7351	0.7816	0.6939	3072,128
20	2400	5.000e-04	0.7356	0.7859	0.6913	3072,128
80	500	5.000e-04	0.7363	0.776	0.7004	3072,128
15	2400	1.000e-03	0.7365	0.7945	0.6864	3072,128
10	2000	5.000e-04	0.7372	0.78	0.6989	3072,128
10	2400	3.000e-04	0.7386	0.7874	0.6954	3072,128
20	2400	1.000e-03	0.7398	0.7948	0.6919	3072,128
60	500	5.000e-04	0.7402	0.7811	0.7034	3072,128
15	2400	3.000e-04	0.7406	0.7902	0.6969	3072,128
25	2400	1.000e-03	0.7425	0.7995	0.6931	3072,128
20	2400	3.000e-04	0.7437	0.7969	0.6972	3072,128
20	2000	5.000e-04	0.745	0.792	0.7034	3072,128
25	2400	3.000e-04	0.7451	0.7986	0.6983	3072,128
30	2400	1.000e-03	0.7467	0.8123	0.691	3072,128
30	2000	5.000e-04	0.7486	0.7997	0.7037	3072,128
30	2400	3.000e-04	**0.7495**	0.8044	0.7016	3072,128

Dataset HHTT+TIA

T5 model - HHTT+TIA TRNS trénovací dataset						
Počet epoch	Steps per epoch	Learning rate	SAcc val. TRNS	SAcc test TRNS	SAcc val. RECOG	SAcc test RECOG
2	1200	1.000e-03	0.775	0.8015	0.6722	0.7373
3	1200	1.000e-03	0.7732	0.8022	0.6685	0.7347
4	1200	1.000e-03	0.809	0.8289	0.6639	0.7213
5	1200	1.000e-03	0.8182	0.8371	0.6804	0.7406
6	1200	1.000e-03	0.8136	0.8293	0.6621	0.7302
7	1200	1.000e-03	0.8145	0.8271	0.6713	0.7213
3	1800	1.000e-03	0.8209	0.839	0.685	0.7351
4	1800	1.000e-03	0.8274	0.8371	0.6795	0.7443
5	1800	1.000e-03	0.8127	0.8382	0.6602	0.7321
3	1800	1.000e-03	0.8191	0.8408	0.6685	0.7347
4	1800	1.000e-03	0.8072	0.8334	0.6593	0.7347
5	1800	1.000e-03	0.8136	0.8378	0.6713	0.7317
3	1800	5.000e-04	0.8237	0.8304	0.663	0.7302
4	1800	5.000e-04	0.831	0.8341	0.6713	0.7217
5	1800	5.000e-04	0.8375	0.8434	0.6841	0.7406
6	1800	5.000e-04	0.8301	0.8397	0.6777	0.7391
4	1800	5.000e-04	0.832	0.836	0.6804	0.7391
5	1800	5.000e-04	0.8228	0.8327	0.6749	0.7369
6	1800	5.000e-04	0.8237	0.8423	0.6758	0.738
7	1800	5.000e-04	0.8292	0.8289	0.6777	0.7332
4	2000	5.000e-04	0.8255	0.8353	0.6685	0.738
5	2000	5.000e-04	0.82	0.8408	0.674	0.7362

6	2000	5.000e-04	0.8237	0.8382	0.6713	0.7332
4	2400	5.000e-04	0.8182	0.8412	0.6788	0.7436
4	2400	3.000e-04	0.8209	0.8338	0.6694	0.7306
5	2400	3.000e-04	0.8375	0.8397	0.6878	0.7351
6	2400	3.000e-04	0.8338	0.843	0.674	0.7365
50	100	1.000e-03	0.8283	0.8341	0.6749	0.7295
50	100	1.000e-03	0.8255	0.839	0.6777	0.7388
4	1800	1.000e-03	0.82	0.836	0.6979	0.7429
5	1800	1.000e-03	0.82	0.8286	0.6933	0.7399
20	100	1.000e-03	0.8356	0.8386	0.6804	0.7384
30	100	1.000e-03	0.8191	0.8297	0.6703	0.7295
40	100	1.000e-03	0.8375	0.8397	0.6749	0.7391
50	100	5.000e-04	0.8329	0.8401	0.6878	0.731
50	100	5.000e-04	0.8283	0.8397	0.686	0.7388
30	500	1.000e-03	0.8246	0.8356	0.6749	0.725
15	500	1.000e-03	0.8053	0.8412	0.6648	0.7391
20	500	1.000e-03	0.831	0.8419	0.6795	0.7388
25	500	1.000e-03	0.8338	0.8367	0.6795	0.7417
4	2200	1.000e-03	0.8173	0.8315	0.6694	0.7347
3	2200	1.000e-03	0.8292	0.8449	0.6832	0.7406
2	2200	1.000e-03	0.8173	0.8364	0.6722	0.7336

T5 model - HHTT+TIA TRNS+RECOG trénovací dataset						
Počet epoch	Steps per epoch	Learning rate	SAcc val. TRNS	SAcc test TRNS	SAcc val. RECOG	SAcc test RECOG
2	1200	1.000e-03	0.8118	0.8275	0.686	0.7447
3	1200	1.000e-03	0.8053	0.8301	0.6786	0.7421
4	1200	1.000e-03	0.8099	0.836	0.6823	0.7503
5	1200	1.000e-03	0.8173	0.833	0.6942	0.7406
6	1200	1.000e-03	0.8237	0.8345	0.6795	0.751
7	1200	1.000e-03	0.8099	0.8304	0.6814	0.7451
3	1800	1.000e-03	0.8145	0.8416	0.6786	0.7592
4	1800	1.000e-03	0.8072	0.8356	0.6841	0.7499
5	1800	1.000e-03	0.8237	0.8275	0.686	0.7429
3	1800	1.000e-03	0.8072	0.8271	0.6933	0.7414
4	1800	1.000e-03	0.8182	0.8226	0.6933	0.7395
5	1800	1.000e-03	0.8145	0.8301	0.6924	0.7466
3	1800	5.000e-04	0.831	0.8271	0.6933	0.7443
4	1800	5.000e-04	0.8136	0.8364	0.685	0.7447
5	1800	5.000e-04	0.8228	0.8312	0.6887	0.7462
6	1800	5.000e-04	0.8255	0.8278	0.6878	0.7455
4	1800	5.000e-04	0.8081	0.8301	0.685	0.7429
5	1800	5.000e-04	0.8099	0.8356	0.6915	0.7481
6	1800	5.000e-04	0.8154	0.8367	0.6887	0.7503
7	1800	5.000e-04	0.8182	0.8356	0.6869	0.7481
4	2000	5.000e-04	0.809	0.833	0.6814	0.7458
5	2000	5.000e-04	0.8301	0.8297	0.697	0.7425
6	2000	5.000e-04	0.8118	0.8364	0.6823	0.7492
4	2400	5.000e-04	0.8191	0.833	0.6896	0.7484
4	2400	3.000e-04	0.8173	0.833	0.6814	0.7503
5	2400	3.000e-04	0.8237	0.8323	0.686	0.7429
6	2400	3.000e-04	0.8145	0.8289	0.686	0.7469
50	100	1.000e-03	0.8356	0.8312	0.7034	0.7481
50	100	1.000e-03	0.8182	0.8275	0.6933	0.7406
4	1800	1.000e-03	0.8044	0.826	0.6713	0.7421
5	1800	1.000e-03	0.8118	0.833	0.6841	0.7495
20	100	1.000e-03	0.82	0.8404	0.6942	0.7529
30	100	1.000e-03	0.832	0.8289	0.6951	0.7484
40	100	1.000e-03	0.8264	0.8319	0.6795	0.7447
50	100	5.000e-04	0.8163	0.8282	0.6905	0.7481
50	100	5.000e-04	0.8035	0.8197	0.6814	0.738
30	500	1.000e-03	0.8127	0.8323	0.6896	0.7532
15	500	1.000e-03	0.8228	0.8334	0.6924	0.7451
20	500	1.000e-03	0.8219	0.8245	0.6997	0.7466
25	500	1.000e-03	0.8118	0.8297	0.6814	0.7481
4	2200	1.000e-03	0.8062	0.8282	0.6841	0.7458
3	2200	1.000e-03	0.8053	0.833	0.685	0.7447
2	2200	1.000e-03	0.8108	0.8252	0.6722	0.7317