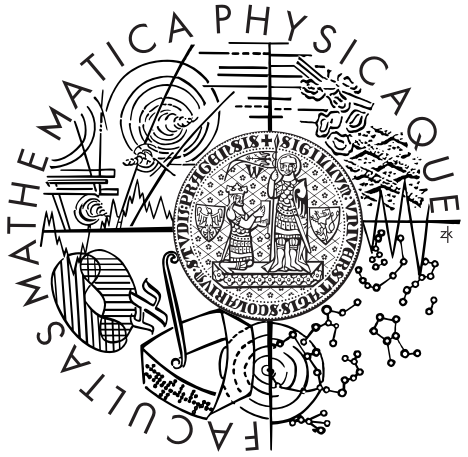
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Matouš Macháček

Metriky pro optimalizaci modelů strojového překladu

Ústav formální a aplikované lingvistiky

Vedoucí bakalářské práce: RNDr. Ondřej Bojar Ph.D.
Studijní program: Informatika
Studijní obor: Obecná informatika

Praha 2012

Tímto bych chtěl poděkovat vedoucímu bakalářské práce RNDr. Ondřejovi Bojarovi Ph.D. za poskytnuté rady a trpělivost při psaní této práce. Také bych chtěl poděkovat Adéle Rusňákové za jazykovou korekturu.

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V Praze dne 25. května 2012 Podpis autora

Název práce: Metriky pro optimalizaci modelů strojového překladu

Autor: Matouš Macháček

Ústav: Ústav formální a aplikované lingvistiky

Vedoucí bakalářské práce: RNDr. Ondřej Bojar Ph.D., UFAL

Abstrakt: Moderní automatické překladové systémy používají takzvaný loglineární model, který skládá dohromady více dílčích modelů a pomocí nich predikuje pravděpodobnost překladu dané věty. Každý dílčí model má v loglineárním modelu svojí váhu. Tyto váhy jsou dnes obecně optimalizovány na skóre automatické metriky BLEU, přestože jsou k dispozici i jiné metriky, z nichž některé korelují lépe s lidskými anotátory než metrika BLEU. V této práci zkoumáme různé metriky (PER, WER, CDER, TER, BLEU a SemPOS) z hlediska korelace s lidskými anotátory. Hlouběji se věnujeme metrice SemPOS a navrhujeme některé její aproximace a varianty. Uvedené metriky jsme použili v metodě MERT při optimalizaci překladového systému z angličtiny do češtiny a zkoumali jsme, jak optimalizování na různé automatické metriky ovlivní výslednou kvalitu systému. V rámci této práce jsme také některé metriky implementovali pro použití v metodě MERT.

Klíčová slova: strojový překlad, automatická metrika, optimalizace modelů, zpracování přirozeného jazyka

Title: Metrics for Optimizing Statistical Machine Translation

Author: Matouš Macháček

Department: Institute of Formal and Applied Linguistics

Supervisor: RNDr. Ondřej Bojar Ph.D., UFAL

Abstract: State-of-the-art MT systems use so called log-linear model, which combines several components to predict the probability of the translation of a given sentence. Each component has its weight in the log-linear model. These weights are generally trained to optimize BLEU, but there are many alternative automatic metrics and some of them correlate better with human judgments than BLEU. We explore various metrics (PER, WER, CDER, TER, BLEU and SemPOS) in terms of correlation with human judgments. Metric SemPOS is examined in more detail and we propose some approximations and variants. We use the examined metrics to train Czech to English MT system using MERT method and explore how optimizing toward various automatic evaluation metrics affects the resulting model.

Keywords: machine translation, automatic metric, optimization, natural language processing

Obsah

1. Úvod

Tato práce porovnává automatické metriky pro měření kvality překladu z angličtiny do češtiny. Cílem je nalézt takovou metriku, která dobře koreluje s lidskými anotátory a zároveň je vhodná pro automatickou optimalizaci strojových překladačů.

1.1 Automatické metriky

Strojový překlad patří mezi nejžádanější úlohy zpracování přirozeného jazyka. Pro zvyšování kvality strojových překladačů je potřeba tuto kvalitu nějak měřit. Pro jednorázové hodnocení systému se může použít lidská anotace: anotátoři dostanou vždy překlad věty a ohodnotí kvalitu překladu věty podle zadaných kritérií. Tento způsob je však drahý a pomalý. Navíc je velmi nevhodný pro vyhodnocování kvality během vývoje překladače, kdy je potřeba často a opakovaně vyhodnocovat velmi podobné překlady. U lidských anotátorů je totiž velmi těžké zaručit konzistentní hodnocení mezi různými anotátory a v různém čase.

Lepší přístup je proto testovací sadu vět, na které se kvalita vyhodnocuje, přeložit nejprve prostřednictvím lidských překladatelů a vytvořit tak referenční překlad. Poté se použije nějaká automatická metrika, která porovná překlad systému s referenčním překladem. Hlavní výhodou tohoto přístupu je rychlost a možnost častého vyhodnocování překladu.

Použitá metrika by měla v ideálním případě hodnotit podobnost hypotézy a reference tak, jak by to hodnotil člověk. Různé metriky však tuto podobnost hodnotí různě dobře. V první části této práce proto počítáme korelaci různých metrik s lidskými anotátory a hledáme nejlépe korelující metriku.

Franz J. Och [22] navrhl použít automatické metriky nejenom při evaluaci, ale také při samotném trénování statistického překladače: Používá je pro nalezení některých parametrů statistického modelu a hodnotu metriky používá jako cílovou funkci optimalizace. V druhé části této práce zkoumáme použití různých metrik při optimalizaci strojových překladačů.

1.2 Předchozí práce

Z hlediska korelace s lidskými anotátory byly metriky porovnávány mnohokrát. Pro angličtinu to bylo třeba v práci [16] nebo v každoroční soutěži automatických metrik během workshopu WMT[1]([8], [11], [9] a [12]). Pro češtinu byly metriky porovnávány například v [18].

Pro optimalizaci strojových překladačů se obecně používá metrika BLEU [24]. Pokusy s jinými metrikami byly provedeny například v [13] nebo v rámci šestého workshopu WMT [12]. V obou případech se však jednalo o optimalizace překladačů do angličtiny. V této práci proto zkoumáme použití automatických metrik pro optimalizaci anglicko-českého statistického překladače.

[1] Workshop on Statistical Machine Translation

1.3 Členění práce

Následující kapitoly jsou rozděleny takto: V kapitole 2 seznamujeme čtenáře se základy statistického strojového překladu a zejména s metodou MERT používanou pro optimalizaci modelu strojového překladu. V kapitole 3 definujeme a popisujeme metriky zkoumané v této práci. Metrice SemPOS se hlouběji věnujeme v kapitole 4. Navrhujeme zde některé aproximace výpočtu této metriky a varianty vzorce pro výpočet celkového skóre metriky. V kapitole 5 popisujeme metodiku experimentů, použitá data a prezentujeme dosažené korelace metrik s lidskými anotátory. Vhodnost různých metrik pro optimalizaci statistických modelů zkoumáme v kapitole 6. V kapitole 7 stručně popisujeme některé implementační záležitosti.

2. Statistický strojový překlad a metoda MERT

První teoretické počátky statistického strojového překladu sahají do roku 1949. Prakticky se statistický překlad začíná výrazně prosazovat až v posledních dvaceti letech. V tomto období totiž začíná být k dispozici dostatečný výpočetní výkon, což byla před tím hlavní překážka.

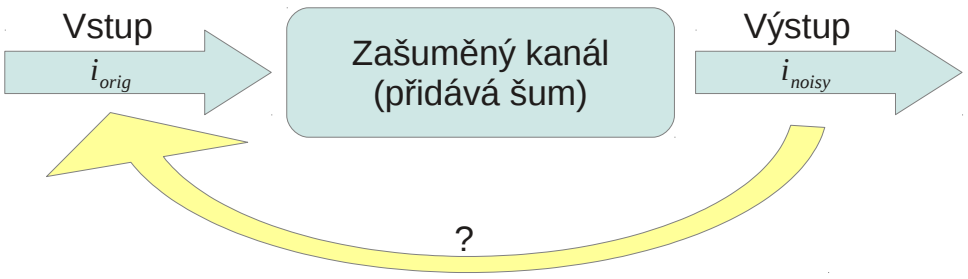
Hlavní výhoda statistických překladačů spočívá v tom, že se dokáží samy naučit překládat z jednoho jazyka do druhého, pokud jim dodáme dostatečné množství dvojjazyčných textů. Na rozdíl od pravidlově založených strojových překladačů není potřeba zaměstnávat lingvisty, kteří by ručně psali pravidla. Tato pravidla navíc většinou bývají jazykově specifická a pro statistické překladače z toho plyne další výhoda: statistický překladač obecně není vytvářen pro konkrétní dvojici jazyků, mezi kterými chceme překládat.

Pokud chceme například překládat z češtiny do angličtiny, natrénujeme překladač na česko-anglických dvojjazyčných datech. Stejný překladač pak můžeme použít i pro překlad jiných jazyků. Jediné, co musíme udělat, je překladač na požadovaný směr překladu natrénovat.

Cílem této práce není seznámit čtenáře s problematikou statistického strojového překladu, avšak pro pochopení metody MERT a využití automatických metrik při trénování statistických překladačů popíšeme stručně v následujících sekcích hlavní principy.

2.1 Zašuměný kanál

Termín Zašuměný kanál (v angličtině Noisy Channel) pochází z teorie informace. Jde o obecný princip, který se používá nejen v lingvistických aplikacích: například rozpoznávání znaků (OCR) nebo rozpoznávání hlasu. Termín popíšeme nejdříve obecně a pak se vrátíme k jeho použití ve statistickém strojovém překladu.



Obrázek 2.1: Úloha zašuměného kanálu: známe zašuměný výstup i_{noisy} a rádi bychom znali původní vstup i_{orig}.

Zašuměný kanál si můžeme představit jako kanál, kterým proudí informace. Na jedné straně je vstup a na druhé výstup. Informace, která na vstupu vstoupí

do kanálu, vystoupí na výstupu. Protože se ale jedná o kanál zašuměný, informace na výstupu není stejná jako na vstupu a obsahuje navíc šum.

Pokud získáme na výstupu nějakou informaci i_{noisy}, nastává přirozená otázka, jaká byla ve skutečnosti původní informace na vstupu i_{orig}. Pro daný výstup ze zašuměného kanálu však může být více vstupů, které zašuměný kanál převedl na daný výstup. Proto hledáme takový vstup, který má nejvyšší pravděpodobnost při daném výstupu. Matematicky to vyjádříme takto:

$$i_{orig} = \arg\max_i P(i \mid i_{noisy}) \tag{2.1}$$

Použitím Bayesovy věty dostaneme:

$$i_{orig} = \arg\max_i \frac{P(i_{noisy} \mid i)P(i)}{P(i_{noisy})} \tag{2.2}$$

Jmenovatel ve zlomku je však konstantní. Dělení konstantou je monotónní funkce a jmenovatele tedy můžeme ve funkci arg max vynechat:

$$i_{orig} = \arg\max_i P(i_{noisy} \mid i)P(i) \tag{2.3}$$

Tento vzorec bychom mohli označit jako základní vzorec zašuměného kanálu. Rozděluje nám totiž problém do dvou menších problémů, které se snadněji a daleko přirozeněji modelují:

- $P(i_{noisy} \mid i)$ – Toto je pravděpodobnostní rozdělení zašuměného výstupu pro daný vstup. Pokud známe povahu zašuměného kanálu nebo pokud máme dostatečné množství pozorování, můžeme toto rozdělení snadno modelovat. V různých aplikacích zašuměného kanálu je tento model nazýván rozdílně.
- $P(i)$ – Toto je pouze pravděpodobnostní rozdělení různých vstupů. Tento model je nezávislý na zašuměném kanálu (to je další výhoda). Většinou ho nazýváme *jazykový model*.

2.1.1 Zašuměný kanál ve statistickém strojovém překladu

Jako základní jednotka překladu se používá věta. Označme si tedy překládanou větu jako f (z anglického „French“ nebo „foreign“). Pro věty v cílovém jazyce budeme používat varianty písmena e („English“).

Úlohou statistického překladu je nalézt větu e_{best}, která je nejpravděpodobnějším překladem dané věty f.

$$e_{best} = \arg\max_e P(e \mid f) \tag{2.4}$$

Ve strojovém statistickém překladu se aplikuje model zašuměného kanálu podobně jako v jiných případech. Na první pohled však nemusí být zřejmé, co ve statistickém překladu považujeme za zašuměný kanál. Ten je totiž v tomto případě spíše uměle vytvořený. Pro lepší pochopení si nejdříve uveďme citát Warrena Weavera [29] z roku 1949:

> „Mám před sebou text, který je napsán v ruštině, ale budu předstírat, že je ve skutečnosti napsaný v angličtině a že byl pouze zakódován do nějakých podivných symbolů. Jediné, co musím udělat, je vyluštit kód, abych získal informaci obsaženou v textu.“

Zašuměným kanálem bude tedy v tomto případě ono smyšlené zakódování do podivných symbolů neboli smyšlený překlad z angličtiny do ruštiny.

Základní vzorec strojového statistického překladu [7] získáme aplikací základního vzorce zašuměného kanálu:

$$e_{best} = \arg\max_{e} P(f \mid e)P(e) \tag{2.5}$$

Bayesův vzorec nám úlohu rozdělil na jazykový model $P(e)$, který říká, jak moc je věta e pravděpodobná v cílovém jazyce, a překladový model $P(f \mid e)$, který říká, jak moc je pravděpodobný překlad věty e na větu f. Všimněte si, že překladový model má opačný směr než směr, ve kterém ve skutečnosti potřebujeme překládat.

Dalo by se namítnout, že získaní parametrů statistického modelu $P(f \mid e)$ je stejně těžký úkol jako získání parametrů modelu $P(e \mid f)$ (rovnice (2.4), ze které jsme vycházeli). Nastává otázka, zda-li jsme si použitím modelu zašuměného kanálu vlastně pomohli.

Problém je, že neznáme skutečné pravděpodobnosti a používané modely tyto pravděpodobnosti neaproximují dostatečně dobře. Tím, že navíc použijeme jazykový model, dokážeme lépe vybírat gramaticky správné věty. Pro trénování parametrů jazykového modelu nám navíc stačí pouze monolingvní data a těch je obvykle mnohem více než dat bilingvních. Z praktického hlediska je tedy provedená dekompozice jistě užitečná.

2.2 Loglineární model

Franz J. Och [23] upozorňuje ve svém článku na několik problémů vzorce (2.5). Kombinace jazykového a překladového modelu je optimální, pouze pokud jsou použity skutečné pravděpodobnosti. Používané modely a trénovací metody však zatím poskytují pouze slabou aproximaci skutečných pravděpodobností. V praxi se navíc prokázaly další skutečnosti:

- Použití přímého překladového modelu místo opačného vykazuje srovnatelné výsledky.
- Jazykový a překladový model nemusejí být stejně dobré. Překlad můžeme zlepšit například zvýšením váhy jazykového modelu.
- Při překladu můžeme kromě jazykového a překladového modelu použít další komponenty.

V modelu zašuměného kanálu však takové úpravy nelze teoreticky ospravedlnit. Navíc ani neexistuje přímočarý a elegantní způsob, jak do modelu zašuměného kanálu přidat další komponenty.

Jako alternativu k zašuměnému kanálu navrhuje Och modelovat pravděpodobnost $P(e \mid f)$ přímo použitím principu maximální entropie. V tomto přístupu definujeme množinu funkcí $h_m(e, f), m = 1, \ldots, M$. Ke každé takové funkci máme její váhu $\lambda_m, m = 1, \ldots M$. Pravděpodobnost překladu je pak přímo modelována vzorcem:

$$P(e \mid f) = p_{\lambda_1^M}(e \mid f) = \frac{\exp\left(\sum_{m=1}^{M} \lambda_m h_m(e, f)\right)}{\sum_{e'} \exp\left(\sum_{m=1}^{M} \lambda_m h_m(e', f)\right)} \tag{2.6}$$

Tento model (dále v textu ho budeme nazývat loglineární model) skládá M různých komponent dohromady. Každá komponenta je reprezentována jednou funkcí $h_m(e, f)$, která předpovídá, jak moc je věta e dobrým překladem věty f z určitého pohledu. Pro tuto funkci používáme dále v textu pojem charakteristická funkce.

Při hledání nejpravděpodobnějšího překladu můžeme ze stejného důvodu jako v rovnici (2.3) vynechat jmenovatele. Můžeme také vynechat funkci exp, protože je rostoucí a nemá vliv na výsledné pořadí.

$$\begin{aligned} e_{best} &= \arg\max_e \frac{\exp\left(\sum_{m=1}^{M} \lambda_m h_m(e, f)\right)}{\sum_{e'} \exp\left(\sum_{m=1}^{M} \lambda_m h_m(e', f)\right)} && (2.7) \\ &= \arg\max_e \exp\left(\sum_{m=1}^{M} \lambda_m h_m(e, f)\right) && (2.8) \\ &= \arg\max_e \sum_{m=1}^{M} \lambda_m h_m(e, f) && (2.9) \end{aligned}$$

Pěknou vlastností loglineárního modelu je fakt, že model zašuměného kanálu je jeho speciálním případem. Položme totiž $M = 2$, $h_1(e, f) = \log P(e \mid f)$, $h_2(e, f) = \log P(e)$ a $\lambda_1 = \lambda_2 = 1$. Pak dosazením do rovnice (2.8) dostáváme:

$$\begin{aligned} e_{best} &= \arg\max_e \exp\left(\log P(e \mid f) + \log P(e)\right) && (2.10) \\ &= \arg\max_e \exp\left(\log P(e \mid f)\right) \exp\left(\log P(e)\right) && (2.11) \\ &= \arg\max_e P(e \mid f) P(e) && (2.12) \end{aligned}$$

Tím jsme se dostali k základní rovnici strojového překladu (2.5).

V loglineárním modelu už není problém provádět úpravy popsané na začátku kapitoly:

- Obrácený překladový model můžeme bez problému nahradit přímým modelem. Můžeme dokonce použít oba dva a zkoumat, zda to pomůže kvalitě překladu.
- Jazykovému a překladovému modelu můžeme nastavit různé váhy.
- Můžeme libovolně přidávat další charakteristické funkce, které nám pomohou s vybíráním nejlepšího překladu.

2.3 MERT - Minimum Error Rate Training

Při použití loglineárního modelu nastávají dva základní problémy:

- **modelování** - návrh vhodných charakteristických funkcí, které dobře předpovídají, jak dobrý je překlad věty f na větu e.
- **optimalizace** - nastavení vhodných hodnot parametrů λ_1^M tak, aby překladač překládal co nejlépe.

V této části se budeme zabývat druhým problémem, tedy jak nastavit váhy dílčích modelů tak, aby loglineární model vybíral nejlepší překlady.

Standardní kritérium pro optimalizaci vah loglineárních modelů je kritérium odvozené z principu maximální entropie:

$$\hat{\lambda}_1^M = \arg\max_{\lambda_1^M} \left\{ \sum_{s=1}^{S} \log p_{\lambda_1^M}(e_s \mid f_s) \right\} \tag{2.13}$$

Tento optimalizační problém má velmi pěkné vlastnosti: existuje jediné globální optimum a existují algoritmy, které konvergují k tomuto optimu.

Franz J. Och však upozorňuje [22], že i když se dají získat pomocí této optimalizační metody dobré výsledky, není žádný důvod domnívat se, že rovnice (2.13) hledá optimální parametry vzhledem ke kvalitě překladu.

Navrhuje proto hledat takové parametry λ_1^M, aby počet chyb v překladu nějakého vyčleněného korpusu $F = \{f_i;\ i \in I\}$ byl co nejnižší[1]. Pokud máme ke korpusu F referenční překlad $R = \{r_i;\ i \in I\}$, můžeme měřit počet chyb překladu pomocí automatické metriky $Err(C, R)$. Hodnotu metriky pro přeložený korpus tedy budeme minimalizovat[2]:

$$\hat{\lambda}_1^M = \arg\min_{\lambda_1^M} \left\{ Err(\hat{E}(F, \lambda_1^M), R) \right\} \tag{2.14}$$

kde $\hat{E}(F, \lambda_1^M)$ je korpus F přeložený s váhami λ_1^M a $\hat{e}(f, \lambda_1^M)$ je překlad věty f s váhami λ_1^M:

$$\hat{E}(F, \lambda_1^M) = \left\{ \hat{e}(f_i, \lambda_1^M);\ i \in I \right\} \tag{2.15}$$

$$\hat{e}(f, \lambda_1^M) = \arg\max_{e \in C_i} \left\{ \sum_{m=1}^{M} \lambda_m h_m(e \mid f) \right\} \tag{2.16}$$

Kritérium (2.14) však nelze optimalizovat tak snadno jako (2.13):

- Obsahuje operaci argmax (rovnice (2.15)), tudíž není možné spočítat gradient a použít ho v nějaké optimalizační metodě používající gradient.
- Cílová funkce má mnoho lokálních minim, se kterými se optimalizační algoritmus musí vypořádat.

[1] Odtud pochází název Minimum Error Rate Training.

[2] Předpokládáme, že hodnota metriky klesá s rostoucí kvalitou překladu. Pokud by to bylo obráceně, budeme maximalizovat.

2.3.1 Optimalizační algoritmus

Pro hledání minima funkce (2.14) se nedá použít žádná optimalizační metoda používající gradient. Franz J. Och proto navrhl [22] použít Powellův algoritmus [25].

Tento algoritmus dostane na vstupu výchozí bod a množinu vektorů (směrů). Navíc potřebuje metodu, která optimalizuje danou funkci podél přímky. Powellův algoritmus začíná ve výchozím bodě. V každé iteraci pak generuje nějaký lepší odhad: Pro každý vektor hledá minimum na přímce procházející aktuálním bodem ve směru daného vektoru. Nejmenší z těchto nalezených minim je pak použito jako nový odhad polohy minima, ze kterého se vychází v další iteraci. Algoritmus je formálně popsán v Algoritmu 1.

Algorithm 1 Powellův algoritmus

```
function POWELL(StartPoint, Directions)
    repeat
        StartValue ← FUNCTION(StartPoint)
        BestPoint ← StartPoint
        BestValue ← StartValue
        for all Direction ∈ Directions do
            Point ← LINEOPTIMIZE(StartPoint, Direction)
            Value ← FUNCTION(Point)
            if Value ≤ BestValue then
                BestPoint ← Point
                BestValue ← Value
            end if
        end for
        StartPoint ← BestPoint
    until StartValue − BestValue ≥ Kappa
    return StartPoint
end function
```

Jako směry, ve kterých se hledá, se používají například osy vektorového prostoru. (To v našem případě znamená, že se při optimalizaci na přímce zafixují všechny parametry λ kromě jednoho, který optimalizujeme.) Implementace metody MERT ve statistickém překladači Moses navíc kromě základních os optimalizuje také podle náhodných vektorů.

Pro optimalizování obecné funkce podél přímky se používají například mřížkové (anglicky „grid-based“) metody. Jejich problém spočívá v tom, že pokud zvolíme vzdálenost zkoušených bodů malou, trvá výpočet dlouho, a pokud velkou, můžeme snadno minout minimum. V našem optimalizačním problému však můžeme využít vlastnosti loglineárního modelu a hledat minimum na přímce efektivně a přesně. Tím se zabývá následující podsekce.

2.3.2 Efektivní hledání minima na přímce

Franz J. Och navrhuje [22] v rámci metody MERT efektivní algoritmus pro hledání minima na přímce. V této části tedy omezíme prostor hledání optimalizačního problému (2.14) na přímku danou rovnicí

$$\lambda(\gamma) = \lambda_1^M + \gamma \cdot d_1^M \tag{2.17}$$

kde λ_1^M je startovní bod a vektor d_1^M určuje směr přímky. Takto upravený optimalizační problém můžeme formálně zapsat takto:

$$\hat{\gamma} = \arg\min_{\gamma \in \mathbb{R}} \left\{ Err(\hat{E}(F, \lambda(\gamma)), R) \right\} \tag{2.18}$$

Funkci (2.16), která vybírá nejlepší překlad, můžeme po dosazení bodů přímky upravit následujícím způsobem:

$$\begin{aligned}
\hat{e}(f, \lambda(\gamma)) &= \arg\max_{e \in C} \left\{ \sum_{m=1}^{M} \lambda(\gamma)_m h_m(e \mid f) \right\} && (2.19)\\
&= \arg\max_{e \in C} \left\{ \sum_{m=1}^{M} (\lambda_m h_m(e \mid f) + \gamma \cdot d_m h_m(e \mid f)) \right\} && (2.20)\\
&= \arg\max_{e \in C} \left\{ \sum_{m=1}^{M} \lambda_m h_m(e \mid f) + \gamma \cdot \sum_{m=1}^{M} d_m h_m(e \mid f) \right\} && (2.21)\\
&= \arg\max_{e \in C} \{ t(e, f) + \gamma \cdot m(e, f) \} && (2.22)
\end{aligned}$$

kde funkce $t(\cdot)$ a $m(\cdot)$ nezávisí na hodnotě γ a pro danou větu f a hypotézu c jsou konstantní. To znamená, že pro každou hypotézu e máme jednu přímku ve tvaru

$$y_e(\gamma) = t(e, f) + \gamma \cdot m(e, f) \tag{2.23}$$

Funkce

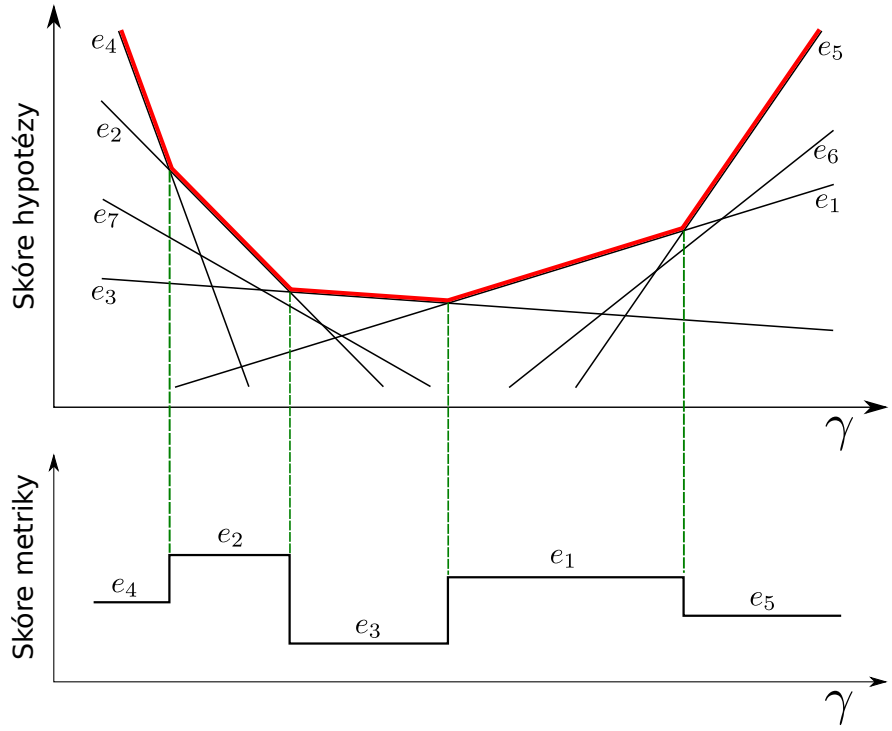
$$g(\gamma) = \max_{e \in C} \{ t(e, f) + \gamma \cdot m(e, f) \} \tag{2.24}$$

která z těchto přímek vybere nejvyšší hodnotu, je po částech lineární (viz obrázek 2.2) a pro každou takovou lineární část je funkce g rovna funkci $y_{e'}$ pro nějakou hypotézu $e' \in C$. To jinými slovy znamená, že na každém takovém intervalu má nejvyšší skóre nějaká hypotéza e'. Tyto intervaly spolu s nejlepší hypotézou můžeme snadno nalézt.

Hraniční body intervalů, na kterých se nemění nejlepší hypotéza, tvoří dělení. Pokud takové dělení vypočítáme pro každou větu korpusu $f \in F$, můžeme zkonstruovat společné poddělení. Pro každý interval tohoto zjemněného dělení známe nejlépe skórující překlad korpusu F a můžeme spočítat hodnotu automatické metriky pro tento překlad. Nakonec z intervalu, který představuje přeložený korpus nejlépe ohodnocený automatickou metrikou, vybereme novou hodnotu $\hat{\gamma}$. Tím získáme řešení optimalizačního problému (2.18).

2.3.3 Aproximace použitím n-best listu

Pokud chceme během optimalizace znát hodnotu cílové funkce, tedy skóre metriky, musíme nejprve přeložit trénovací korpus s danými vahami a překlad ohodnotit automatickou metrikou. Překlad korpusu je však výpočetně náročný, proto se



Obrázek 2.2: Graf skóre hypotéz a hodnoty metriky. Ve vrchní části jsou znázorněny přímky odpovídající hypotézám. Červeně je pak znázorněna funkce, která z těchto přímek vybírá maximum. V dolní části grafu je pak na intervalech, na kterých vyhrává konkrétní hypotéza, znázorněna hodnota metriky.

místo toho používá aproximace pomocí n-best listu: Před zahájením optimalizace vyprodukuje dekodér s počátečními vahami celou množinu kandidátských překladů trénovacího korpusu, takzvaný n-best list. To je prakticky stejně výpočetně náročné jako obyčejný překlad (dekodér si během překladu jednoduše pamatuje n nejlepších překladů místo jediného). U každé věty v n-best listu máme navíc i hodnoty charakteristických funkcí.

Když pak potřebuje algoritmus během optimalizace získat hodnotu v nějakém bodě, nespustí se dekodér, ale z n-best listu se vyberou překlady, které s danými vahami dosahují nejvyššího skóre.

Problém však nastane tehdy, pokud se vektor vah během optimalizace dostane daleko od vah, se kterými byl n-best list vyprodukován. V takovém místě totiž můžou být nejlepším překladem věty, které se v počátečním n-best listu vůbec nevyskytují a můžou obsahovat větší počet chyb.

Franz J. Och [22] tento problém řeší tak, že po doběhnutí Powellova algoritmu vyprodukuje pomocí získaných vah nový n-best list a sloučí ho s předchozím n-best listem. Pak pustí optimalizační algoritmus na sloučeném n-best listu znovu. (N-best listu, který vzniká postupným slučováním při každé iteraci, říkáme akumulovaný n-best list.) Po dostatečném počtu iterací se akumulovaným n-best listem pokryje dostatečně velký prostor a nové n-best listy nepřidávají nové překlady. Powellův algoritmus by měl v neměnícím se n-best listu nalézat stejný vektor vah a zůstávat na místě. Tak poznáme, že jsme našli optimum a končíme s hledáním.

3. Automatické metriky

V této kapitole popíšeme metriky, které budeme v této práci zkoumat z hlediska korelace s lidskými anotátory a z hlediska použití při optimalizaci vah loglineárního modelu.

Budeme používat následující značení: pro testovanou větu budeme používat c (z anglického „candidate“) a pro její referenční překlad r. Pokud budeme definovat metriku pro celý testovaný korpus, budeme používat I jako indexovou množinu do množiny testovaných vět $C = \{c_i;\ i \in I\}$ a do množiny referenčních překladů $R = \{r_i;\ i \in I\}$, kde platí, že r_i je referenčním překladem věty c_i pro všechna $i \in I$.

Většina metrik může být definována pro více referenčních překladů. V této práci jsme však měli k dispozici vždy pouze jediný referenční překlad, a proto budeme pro jednoduchost definovat metriky pouze s jedním referenčním překladem.

3.1 BLEU - Bilingual Evaluation Understudy

Metrika BLEU [24] je považována za standardní automatickou metriku pro strojový překlad. Tato metrika vypočítá pro každé $n \in \{1 \dots N\}$ přesnost překladu n-gramů. Tím že používá n-gramy, dokáže metrika správně ohodnotit plynulost hypotézy. Z vypočítaných přesností pak vypočítá geometrický průměr a navíc penalizuje krátké věty:

$$BLEU(C, R) = BP \cdot \exp\left(\frac{1}{N}\sum_{i=1}^{N} \log p_n\right) \tag{3.1}$$

kde N bude v našem případě 4 a p_n je přesnost n-gramů daná vztahem

$$p_n = \frac{\sum_{i \in I} \sum_{\text{n-gram} \in c_i} \min(\text{count}(\text{n-gram}, c_i), \text{count}(\text{n-gram}, r_i))}{\sum_{i \in I} \sum_{\text{n-gram} \in c_i} \text{count}(\text{n-gram}, c_i)} \tag{3.2}$$

a BP je penalta pro krátké hypotézy (brevity penalty) daná vztahem

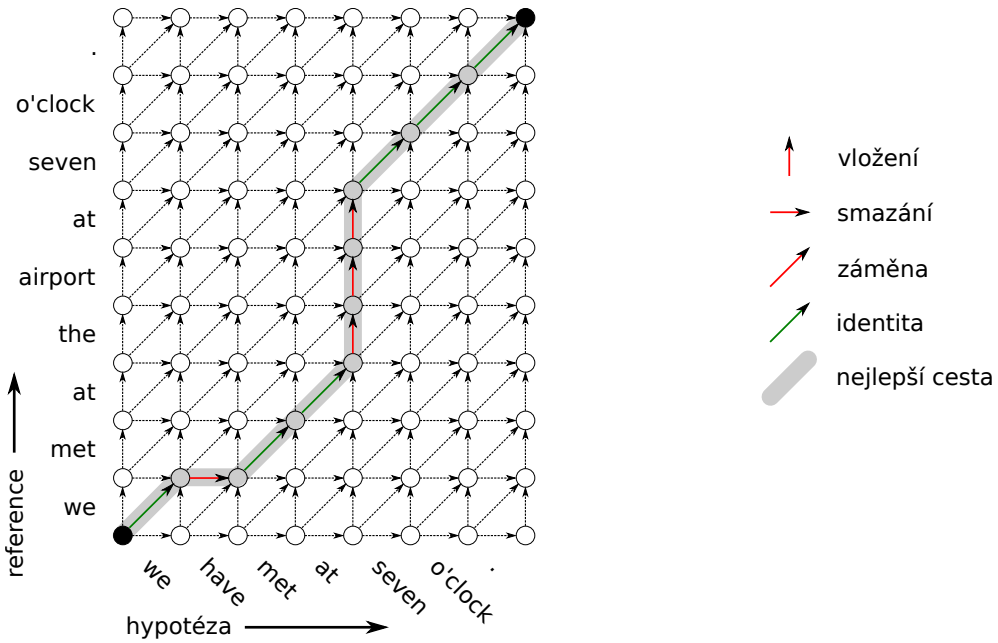
$$BP = \begin{cases} 1 & \text{pokud } |c| > |r| \\ \exp(1 - |r|/|c|) & \text{pokud } |c| \leq |r| \end{cases} \tag{3.3}$$

Tato penalta snižuje skóre krátkým větám za nepřeloženou informaci.

3.2 WER - Word Error Rate

Následující čtyři metriky jsou založeny na editační vzdálenosti. Počítáme v nich počet editací, které je potřeba provést, abychom testovanou větu transformovali na referenční větu.

Nejjednodušší a nejpřirozenější taková metrika je WER [28] (Word Error Rate), která počítá Levenshteinovu vzdálenost mezi testovanou hypotézou a referencí. Dovolené editační úpravy jsou smazání, záměna a přidání slova. Posloupnost takových editačních úprav můžeme znázornit jako cestu v mřížce zarovnání.



Obrázek 3.1: Příklad mřížky zarovnání metriky WER

Příklad takové cesty je na obrázku 3.1. Metrika spočítá minimální počet těchto operací a výsledný počet pak znormalizuje. Formálně tedy:

$$WER(c,r) = \frac{\min_{e \in E(c,r)}(D(e) + S(e) + I(e))}{|r|} \tag{3.4}$$

kde $E(c,r)$ značí množinu všech editačních posloupností, které transformují větu c na větu r, $D(e)$ je počet smazání, $S(e)$ je počet záměn a $I(e)$ je počet vložení v posloupnosti editací e.

Pokud chceme vypočítat hodnotu metriky pro celou množinu vět, sečteme počet editací pro každou větu a součet normalizujeme až nakonec:

$$WER(C,R) = \frac{\sum_{i \in I} \min_{e \in E(c_i,r_i)}(D(e) + S(e) + I(e))}{\sum_{i \in I} |r_i|} \tag{3.5}$$

3.3 TER - Translation Error Rate

Občas se stává, že systémem přeložená věta se od referenčního překladu liší pořadím celých bloků slov. Metrika WER by takovou větu velmi penalizovala, ale člověk by takový překlad často ohodnotil jako kvalitní.

Tento problém se snaží řešit metrika TER [27] (Translation Error Rate), která je velmi podobná metrice WER. Mezi povolené editační úpravy navíc přidává i přesun celých bloků slov. Všechny úpravy, včetně přesunu libovolně velkého bloku o libovolnou vzdálenost, mají stejnou cenu:

$$TER(c,r) = \frac{\min_{e \in E(c,r)}(D(e) + S(e) + I(e) + SH(e))}{|r|} \tag{3.6}$$

kde $E(c,r)$ značí množinu všech editačních posloupností, které transformují větu c na větu r, $D(e)$ je počet smazání, $S(e)$ je počet záměn $I(e)$ je počet vložení a

$SH(e)$ je počet blokových přesunů v posloupnosti editací e. Výpočet metriky na celém testovacím korpusu je dán tímto vzorcem:

$$TER(C,R) = \frac{\sum_{i \in I} \min_{e \in E(c_i,r_i)}(D(e) + S(e) + I(e) + SH(e))}{\sum_{i \in I} |r_i|} \tag{3.7}$$

3.4 CDER - Cover Disjoint Error rate

Metrika CDER [19] se podobně jako TER pokouší vylepšit metriku WER dovolením přesunů celých bloků. Na rozdíl od metriky TER nepočítá počet blokových přesunů, ale zavádí místo toho operaci dlouhého skoku.

V metrice CDER je přidán k základním editačním úpravám dlouhý skok. Při dlouhém skoku se můžeme v mřížce zarovnání přesunout na libovolné místo v testované větě za jednotkovou cenu (V mřížce zarovnání se tedy přesuneme v rámci řádky). Na obrázku 3.2 se nachází mřížka zarovnání s dlouhými skoky. Je vidět, že tímto způsobem zbytečně nepenalizujeme přesuny celých bloků. Formálně je metrika definována takto:

$$CDER(c,r) = \frac{\min_{p \in P(c,r)}(D(p) + S(p) + I(p) + LJ(p))}{|r|} \tag{3.8}$$

kde $P(c,r)$ značí množinu všech korektních cest v mřížce zarovnání, které transformují větu c na větu r, $D(p)$ je počet smazání, $S(p)$ je počet záměn $I(p)$ je počet vložení a $LJ(p)$ je počet dlouhých skoků v cestě p v mřížce zarovnání. Výpočet metriky na celém testovacím korpusu je dán tímto vzorcem:

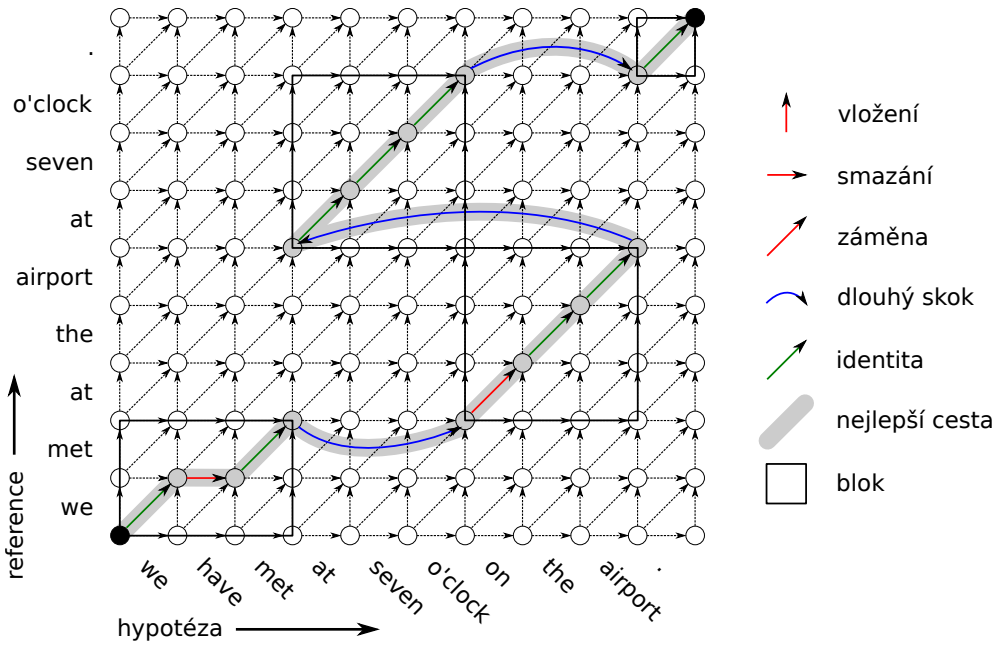
$$CDER(C,R) = \frac{\sum_{i \in I} \min_{p \in P(c_i,r_i)}(D(p) + S(p) + I(p) + LJ(p))}{\sum_{i \in I} |r_i|} \tag{3.9}$$

Nakonec si všimněte, že v metrice CDER není zaručeno, že cesta v mřížce zarovnání bude pokrývat všechna slova testované věty. Některá slova můžou být dokonce pokryta vícekrát. Na druhou stranu jsou pokryta všechna slova reference. Důsledkem tedy je, že metrika bude penalizovat slova z reference, která nejsou v testované větě, ale už nemusí penalizovat za slova, která jsou v testované větě navíc. Pokud bychom řekli, že každé slovo testované věty musí být pokryto právě jednou, měli bychom NP-těžký problém. Bez tohoto omezení lze úlohu řešit dynamickým programováním podobně jako výpočet Levenshteinovy vzdálenosti v metrice WER.

3.5 PER - Position Independent Error Rate

Metriku PER můžeme zařadit také do metrik založených na editační vzdálenosti. Na rozdíl od předchozích tří metrik tato metrika nezohledňuje pořadí slov.

Metrika PER je podobně jako metrika WER definovaná jako počet smazání, záměn a přidání slov, které jsou potřeba pro transformování testované věty na referenční větu. Protože metrika nezohledňuje pořadí slov, můžeme metriku definovat jednoduchým vzorečkem:



Obrázek 3.2: Příklad mřížky zarovnání metriky CDER

$$PER(c, r) = \frac{\max(|r \setminus c|, |c \setminus r|)}{|r|} \tag{3.10}$$

kde r a c jsou multimnožiny slov z testované a referenční věty.

3.6 SemPOS - Semantic Part of Speech Overlapping

Metrika SemPOS byla poprvé představena v práci Kamila Kose a Ondřeje Bojara [4]. Tato metrika je inspirovaná sadou metrik pracujících s různými lingvistickými prvky na syntaktické a sémantické úrovni, kterou navrhli J. Giménez a L. Márquez [15]. Jednou z jejich nejlepších metrik byla metrika *Semantic Role Overlapping* (česky „pokrytí sémantickými rolemi“), která brala věty jako množiny slov a jejich sémantických rolí a podobnost mezi těmito množinami počítala pomocí obecné míry podobnosti zvané *Overlapping*, kterou za chvíli definujeme.

Kamil Kos a Ondřej Bojar však místo sémantických rolí, které nejsou definované pro češtinu, použili nástroj TreeX [30] (dříve TectoMT) pro získání sémantického slovního druhu definovaném v [26]. Navíc místo slovních forem použili tektogramatická lemmata (zkráceně *t-lemmata*). Důsledkem těchto změn je, že jsou brána v potaz pouze plnovýznamová slova. Tuto novou metriku pojmenovali *Semantic Part of Speech Overlapping*, zkráceně SemPOS.

Pro výpočet metriky se tedy nejdříve testované i referenční věty věty analyzují na tektogramatickou rovinu, aby se získaly tektogramatická lemmata a jejich sémantické slovní druhy. Pro každý sémantický slovní druh $t \in T$, kde T označuje množinu všech sémantických slovních druhů, pak vypočítáme hodnotu pokrytí:

$$\mathrm{Ovr}_t(c, r) = \frac{\sum_{w \in r} \min(\mathrm{count}_t(t, r), \mathrm{count}_t(w, c))}{\sum_{w \in r} \mathrm{count}_t(w, r)} \tag{3.11}$$

kde $\mathrm{count}_t(w, s)$ označuje počet slov w se sémantickým slovním druhem t ve větě s. Pokud počítáme pokrytí pro celý korpus vět, sečteme podobně jako u předchozích metrik zvlášť jmenovatele a zvlášť čitatele:

$$\mathrm{Ovr}_t(C, R) = \frac{\sum_{i \in I} \sum_{w \in r_i} \min(\mathrm{count}_t(t, r_i), \mathrm{count}_t(w, c_i))}{\sum_{i \in I} \sum_{w \in r_i} \mathrm{count}_t(w, r_i)} \tag{3.12}$$

Výsledná hodnota metriky je pak definována jako aritmetický průměr pokrytí jednotlivých slovních druhů:

$$SemPOS(C, R) = \frac{1}{|T|} \sum_{t \in T} \mathrm{Ovr}_t(C, R) \tag{3.13}$$

Této metrice se hlouběji věnujeme a zkoumáme některé její varianty v kapitole 4.

4. Aproximace a varianty metriky SemPOS

Metrika SemPOS koreluje dobře s lidskými anotátory [4]. Hlavní nevýhoda je její výpočetní složitost, která je způsobena parsováním až na tektogramatickou rovinu, z níž metrika získává tektogramatická lemmata a sémantické slovní druhy. Výpočetní složitost bohužel dělá tuto metriku nevhodnou pro optimalizaci vah v metodě MERT. V podkapitole 4.1 tedy popisujeme různé aproximace pro získání t-lemmat a semposů bez parsování na tektogramatickou rovinu.

J. Giménez a L. Márquez [15] a O. Bojar a K. Kos [4] použili různé vzorce pro výpočet pokrytí[1]. V podkapitole 4.2 popisujeme oba tyto vzorce a navíc uvádíme jeden vzorec vlastní.

Zkombinováním určité aproximační metody a určitého vzorce pokrytí získáme variantu metriky SemPOS. Kvalita těchto variant je zkoumána v části 5.4.

4.1 Aproximace tektogramatické roviny

Rádi bychom získali tektogramatická lemmata a sémantické slovní druhy bez analyzování na tektogramatickou rovinu. Ve všech aproximacích budeme vycházet pouze z morfologického značkování a lemmatizace.

Kromě jednoho případu (viz sekce 4.1.4) budeme aproximovat tektogramatická lemmata použitím obyčejných povrchových lemmat. Pro naprostou většinu plnovýznamových slov se tektogramatické lemma rovná povrchovému lemmatu. Přesto existují některé třídy slov, kde se tektogramatické lemma liší. V těchto případech se t-lemma většinou skládá z povrchového lemmatu hlavního významového slova a z pomocného slova (například „smát_se“).

Aproximace sémantického slovního druhu už tak jednoduchá není. V následujících sekcích popisujeme čtyři varianty aproximace.

4.1.1 Sempos z morfologické značky

Všimli jsme si, že morfologická značka určuje téměř jednoznačně sémantický slovní druh. Použili jsme tedy česko-anglický paralelní korpus CzEng [5] pro vytvoření jednoduchého slovníku, který mapuje morfologické značky na nejčastější sémantický slovní druh, který jsme viděli v korpusu. Některá neplnovýznamová slova ovšem nejsou reprezentována na tektogramatické rovině, a proto mapujeme některé značky i na speciální hodnotu „-“, která říká, že slovo s danou značkou nemá reprezentaci na tektogramatické rovině. Slova s takovou značkou pak nejsou při výpočtu metriky brána v úvahu.

První aproximace tedy spoléhá pouze na tento slovník. Vstupní text je automaticky označkován, morfologické značky jsou deterministicky převedeny na

[1] Ve skutečnosti J. Giménez a L. Márquez [15] publikovali dvě verze svého článku, které byly skoro totožné, ale lišily se vzorcem pro výpočet pokrytí. Zeptali jsme se proto přímo autorů, která verze je správná. Ukázalo se, že O. Bojar a K. Kos, nevěda o dvou verzích článku, použili v [4] vzorec z nesprávné verze článku. Protože i s tímto vzorcem dosáhli dobrých výsledků, rozhodli jsme se, že v této práci zkusíme a porovnáme oba vzorce.

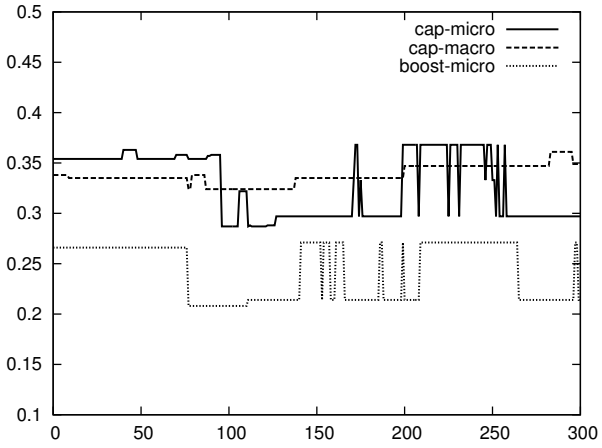
sémantické slovní druhy pomocí vytvořeného slovníku a slova, u kterých byla značka převedena na hodnotu „-“, jsou odstraněna. Přesnost určení semposů pomocí této metody je pro češtinu ve srovnání s neaproximovanou metodou 88,4 %.

V této práci budeme tuto metodu nazývat APPROX.

4.1.2 Zahazování nejčastějších slov

Tektogramatická rovina používá podle definice pouze plnovýznamová slova. Většina pomocných slov se na tektogramatické rovině vyskytuje pouze jako atribut plnovýznamových uzlů a nehraje roli v pokrytí mezi testovanou větou a referencí.

První aproximační metoda rozpoznávala pomocná slova pouze pomocí morfologických značek. Naší snahou je rozpoznat a zahodit všechna pomocná slova. V této aproximační metodě postupujeme stejně jako v první metodě, ale navíc zahazujeme určitý počet nejčastějších slov v jazyku, protože předpokládáme, že taková slova jsou pomocná. Seznam nejčastějších slov jsme získali z české části paralelního korpusu CzEng [5].



Obrázek 4.1: Korelace závisející na délce seznamu slov, které zahazujeme. Tuto závislost jsme počítali se všemi vzorci pokrytí (Viz sekce 4.2).

Zkoušeli jsme různé délky seznamu nejčastějších slov, které zahazujeme. Výsledky tohoto experimentu můžete vidět na obrázku 4.1.2. Je vidět, že se křivka korelace nechová moc pěkně a nelze vysledovat žádný trend. Přesto jsme pro tuto metodu nazývanou APPROX-STOPWORDS určili délku seznamu zahazovaných slov 220.

4.1.3 Omezení množiny sémantických slovních druhů

Všimli jsme si, že různé druhy semposů přispívají k celkové korelaci metriky různě. Jedním z důvodů může být větší či menší přesnost značkování některých slovních tříd, dalším důvodem může být to, že chyby v překladu některých slovních tříd jsou pro lidské anotátory relevantnější.

V tabulce 4.1 jsme spočítali korelace metrik, které počítají pokrytí vždy pouze jednoho sémantického slovního druhu. Na základě této tabulky předpokládáme,

Sempos	Četnost	Min.	Max.	Avg.	Použito
n.pron.def.pers	0.030	0.406	0.800	0.680	*
n.pron.def.demon	0.026	0.308	1.000	0.651	*
adj.denot	0.156	0.143	0.874	0.554	*
adv.denot.ngrad.nneg	0.047	0.291	0.800	0.451	*
adv.denot.grad.nneg	0.001	0.219	0.632	0.445	*
adj.quant.def	0.004	-0.029	0.800	0.393	
n.denot.neg	0.037	0.029	0.736	0.391	
adv.denot.grad.neg	0.018	-0.371	0.800	0.313	
n.denot	0.432	-0.200	0.720	0.280	*
adv.pron.def	0.000	-0.185	0.894	0.262	
adj.pron.def.demon	0.000	0.018	0.632	0.241	
n.pron.indef	0.027	-0.200	0.423	0.112	
adj.quant.grad	0.006	-0.225	0.316	0.079	
v	0.180	-0.600	0.706	0.076	*
adj.quant.indef	0.002	-0.105	0.200	0.052	
adv.denot.ngrad.neg	0.000	-0.883	0.775	0.000	
n.quant.def	0.000	-0.800	0.713	-0.085	

Tabulka 4.1: Sémantické slovní druhy a korelace metriky, která počítá pokrytí pouze jednoho druhu. Pro získání semposů byla použita metoda APPROX a pro výpočet byl použit vzorec pokrytí CAP (Viz sekce 4.2). Korelace byla vypočítána stejně jako v kapitole 5. V nové aproximaci počítáme pokrytí pouze těchto semposů: v, n.denot, adj.denot, n.pron.def.pers, n.pron.def.demon, adv.denot.ngrad.nneg, adv.denot.grad.nneg. V tabulce jsou tyto sémantické slovní druhy označeny hvězdičkou.

že některé druhy semposů zvyšují korelaci celkového pokrytí s lidskými anotátory a některé ji snižují. Proto v této variantě počítáme pokrytí pouze vybrané podmnožiny druhů sempos.

Tato aproximace nazvaná APPROX-RESTR je založena na první aproximaci, ale počítá pokrytí pouze této podmnožiny slovních druhů: v, n.denot, adj.denot, n.pron.def.pers, n.pron.def.demon, adv.denot.ngrad.nneg, adv.denot.grad.nneg.

4.1.4 Použití taggeru pro získání t-lemmat a semposů

Naše poslední aproximační metoda se od předchozích tří metod hodně liší. Používáme sekvenční značkovací algoritmus [14] implementovaný v programu Featurama[2] pro výběr t-lemmatu a semposu. Použili jsme českou stranu korpusu CzEng [5] pro natrénování značkovače. Na každém tokenu má značkovač k dispozici slovní formu, morfologickou značku a povrchové lemma (aktuálního tokenu a předchozích dvou tokenů), aby vybral jeden pár t-lemmatu a semposu z nabízené množiny.

Množina možných párů t-lemmat a semposů je vytvořena následovně: Nejprve je vytvořena množina semposů. Jednoduše použijeme všechny semposy, které jsme viděli v korpusu s danou morfologickou značkou. Poté vytvoříme množinu

[2] `http://sourceforge.net/projects/featurama`

možných t-lemmat. Pro většinu druhů semposu použijeme povrchové lemma jako jediné tektogramatické lemma. Pro sempos „v“ přidáváme k povrchovému lemmatu také t-lemma složené z povrchového lemmatu a nějakého pomocného slova přítomného ve větě („smát_se“). Pro některé další značky semposu přidáváme speciální t-lemmata pro negaci a osobní zájmena („#Neg“,„#PersPron“).

Celková přesnost natrénovaného značkovače na evaluační množině je 94,9 % pro češtinu, což je lepší výsledek na těžším úkolu (protože určujeme i t-lemmata) než je deterministické značkování v podsekci 4.1.1.

4.2 Varianty vzorce pro výpočet pokrytí

Původní vzorec pokrytí definovaný J. Giménezem a L. Márquezem [15] je dán rovnicemi (4.1) a (4.2). První z nich hledí pouze na jeden sémantický druh, druhý pak udává pokrytí pro všechny druhy:

$$\mathrm{Ovr}_t(c,r) = \frac{\sum_{w \in r} \mathrm{count}_t(w,c)}{\sum_{w \in r \cup c} \max(\mathrm{count}_t(w,r), \mathrm{count}_t(w,c)} \tag{4.1}$$

kde c a r značí testovanou a referenční větu a $\mathrm{count}_t(w,s)$ značí počet slov w typu (semposu) t ve větě s. Pro každý typ semposu t, $\mathrm{Ovr}_t(c,r)$ počítá podíl správně přeložených slov typu t. V této práci budeme nazývat tento vzorec BOOST.

Rovnice (4.2) popisuje výpočet pokrytí všech typů:

$$\mathrm{Ovr}(c,r) = \frac{\sum_{t \in T} \sum_{w \in r} \mathrm{count}_t(w,c)}{\sum_{t \in T} \sum_{w \in r \cup c} \max(\mathrm{count}_t(w,r), \mathrm{count}_t(w,c)} \tag{4.2}$$

kde T značí množinu všech typů semposu. Tento vzorec pokrytí budeme nazývat BOOST-MICRO, protože počítá mikro průměr pokrytí jednotlivých typů.

O. Bojar a K. Kos [4] použili mírně odlišný vzorec, v této práci označovaný CAP:

$$\mathrm{Ovr}_t(c,r) = \frac{\sum_{w \in r} \min(\mathrm{count}_t(w,c), \mathrm{count}_t(w,r))}{\sum_{w \in r \cup c} \mathrm{count}_t(w,r)} \tag{4.3}$$

Pro výpočet pokrytí všech typů pak použili běžný makro průměr. Tuto metodu nazýváme CAP-MACRO:

$$\mathrm{Ovr}(c,r) = \frac{1}{|T|} \sum_{t \in T} \mathrm{Ovr}_t(c,r) \tag{4.4}$$

Rozdíl mezi mikro a makro průměrem spočívá v tom, že při použití makro průměru mají všechny typy stejnou váhu bez ohledu na četnost výskytu. Například $\mathrm{Ovr}_{\mathrm{n.denot}}$ a $\mathrm{Ovr}_{\mathrm{adv.denot.grad.nneg}}$ budou mít stejnou váhu, ale přitom položek typu n.denot je daleko více než položek typu adv.denot.grad.nneg. Protože se nám tento fakt zdá nepřirozený, navrhujeme jeden nový vzorec pokrytí nazvaný CAP-MICRO:

$$\mathrm{Ovr}(c,r) = \frac{\sum_{t \in T} \sum_{w \in r} \min(\mathrm{count}_t(w,c), \mathrm{count}_t(w,r))}{\sum_{t \in T} \sum_{w \in r \cup c} \mathrm{count}_t(w,r)} \tag{4.5}$$

Celkem tedy máme tři vzorce pro výpočet celkového pokrytí: BOOST-MICRO (4.2), CAP-MACRO (4.4) a CAP-MICRO (4.5).

5. Korelace metrik s lidskými anotátory

Metriky uvedené v předchozí kapitole zkoumáme ze dvou hlavních hledisek: Z hlediska korelace s lidskými anotátory (tedy jak dobře rozpozná metrika kvalitní překlad) a z hlediska použití při optimalizaci vah statistického modelu (jak rychlý je výpočet metriky, jak rychle optimalizační algoritmus konverguje, zda má optimalizovaná funkce pěkný tvar atd.). V této kapitole zkoumáme metriky z prvního hlediska. Míra korelace s lidskými anotátory není důležitá jenom pro metriky používané pro porovnávání systémů mezi sebou. Při optimalizaci modelu statistického překladače totiž nemůže metrika, která nekoreluje dobře s lidskými anotátory, zaručit, že model s optimalizovanými vahami bude překládat lépe než neoptimalizovaný model.

5.1 Testovací data

Jako testovací data pro měření korelace s lidskými anotátory jsme použili data shromážděná během čtyř ročníků WMT (Workshop on Statistical Machine Translation): WMT08 [8], WMT09 [11], WMT10 [9] a WMT11[1] [12]. V každém ročníku se konala soutěž strojových překladačů, ve které překladače překládaly určitou testovací sadu vět. Podrobnosti o těchto testovacích sadách naleznete v tabulce 5.1.

5.2 Výpočet lidského hodnocení

Pro každou větu z testovací sady tedy máme od každého systému v soutěži jeden překlad. Anotátoři hodnotili tyto překlady tímto způsobem: Dostali vždy maximálně 5 překladů nějaké věty, které měli seřadit od nejlepší po nejhorší. Mohli také dát více překladům stejné pořadí. Takto vytvořené pořadí pak sloužilo jako „simulované porovnávání po párech“: všechny páry systémů v pořadí byly použity jako samostatné porovnání dvou systémů. Lidské skóre překladového systému pak bylo vypočítáno jako procento párů, ve kterých byl daný systém lepší nebo stejně dobrý, ze všech párů, ve kterých se daný systém vyskytl.

Tento způsob výpočtu lidského hodnocení (označovaný jako „$\geq$ others“) jsme se rozhodli použít, protože je to oficiální způsob používaný během soutěží WMT. Jako varianta tohoto výpočtu se používá metrika, která počítá procento párů, ve kterých byl daný systém lepší („$>$ others“).

V poslední době se však objevily námitky proti tomuto způsobu a byly navrženy ([2], [20]) i jiné způsoby výpočtu lidského skóre.

[1] Data z ročníku WMT11 byla použita pouze v části 5.5, v části 5.4 jsme tato data ještě neměli k dispozici.

Workshop	Název sady	Počet vět	Počet systémů
WMT08	nc-test2008	2028	4
WMT08	newstest2008	2051	4
WMT09	newstest2009	2525	6
WMT10	newssyscombtest2010	2034	12
WMT11	newstest2011	3003	10

Tabulka 5.1: Testovací sady použité pro výpočet korelace metrik s lidskými anotátory.

5.3 Výpočet korelace

Zkoumanou metriku jsme vypočítali pro každý systém. V každé sadě jsme tedy měli pro každý systém skóre metriky a lidské skóre. Z těchto dat jsme pak vypočítali Spearmanův korelační koeficient. Protože nenastaly žádné shody, použili jsme pro výpočet korelačního koeficientu vzorec

$$\rho = 1 - \frac{6 \sum_i (p_i - q_i)^2}{n(n^2 - 1)} \tag{5.1}$$

kde p_i je pořadí systému i podle skóre metriky, q_i je pořadí systému i podle lidského skóre a n je počet systémů.

Protože lidská skóre nejsou porovnatelná mezi různými testovacími sadami, spočítali jsme korelační koeficient pro každou sadu zvlášť a pak z nich vypočítali minimální, průměrnou a maximální korelaci.

5.4 Výsledky různých variant SemPOSu

V kapitole 4 jsme popsali několik aproximačních metod a několik vzorců, podle kterých se počítá pokrytí. Každá kombinace aproximační metody a vzorce pro výpočet pokrytí nám dává novou variantu metriky SemPOS. V tabulce 5.2 naleznete dosažené korelace těchto variant.

Pro porovnání s původní metrikou SemPOS jsme do tabulky zařadili i neaproximované varianty, kde jsou t-lemmata a semposy získány pomocí nástroje TreeX. Tyto varianty mají v tabulce v prvním sloupci položku ORIG.

Nejlépe korelující varianta je kombinace APPROX-RESTR a CAP-MACRO.

Pokud budeme hodnotit výkon jednotlivých aproximačních metod, můžeme zobecnit výsledky z tabulky a říct, že metody korelují obecně v tomto pořadí od nejlepší po nejhorší: APPROX-RESTR, TAGGER, ORIG, APPROX-STOPWORDS a APPROX. Metody APPROX-RESTR a TAGGER korelují dokonce lépe než neaproximovaná metoda ORIG. V tomto smyslu se podařil původní cíl aproximovat metriku SemPOS (i za cenu nižší korelace) nad očekávání. Metoda APPROX-STOPWORDS koreluje jen nepatrně lépe než APPROX a vzhledem k nepěknému chování metody v grafu 4.1.2 můžeme tuto metodu do budoucna zavrhnout.

Při hodnocení vzorců pro výpočet pokrytí vyšel nejhůře vzorec BOOST-MICRO. Považujeme proto tento vzorec jako nevhodný pro jakoukoliv metriku založenou na sémantických slovních druzích. Srovnání vzorců CAP-MACRO a CAP-MICRO už není tak jednoznačné, ale přesto je CAP-MACRO o něco lepší. Možným vysvětlením

Aproximace	Vzorec pokrytí	Min.	Max.	Avg.
APPROX-RESTR	CAP-MACRO	0.400	0.800	0.608
TAGGER	CAP-MACRO	0.143	0.800	0.428
ORIG	CAP-MACRO	0.143	0.800	0.423
APPROX-RESTR	CAP-MICRO	0.086	0.769	0.413
TAGGER	CAP-MICRO	0.086	0.769	0.413
ORIG	CAP-MICRO	0.086	0.741	0.406
APPROX-STOPWORDS	CAP-MICRO	0.086	0.790	0.368
APPROX	CAP-MICRO	0.086	0.734	0.354
APPROX-STOPWORDS	CAP-MACRO	0.086	0.503	0.347
APPROX	CAP-MACRO	0.086	0.469	0.338
TAGGER	BOOST-MICRO	0.086	0.664	0.337
ORIG	BOOST-MICRO	-0.200	0.692	0.273
APPROX-STOPWORDS	BOOST-MICRO	-0.200	0.685	0.271
APPROX	BOOST-MICRO	-0.200	0.664	0.266
APPROX-RESTR	BOOST-MICRO	-0.200	0.664	0.266

Tabulka 5.2: Korelace jednotlivých variant metriky SemPOS. První dva sloupce určují použitou kombinaci aproximační metody a vzorce pro výpočet pokrytí. U každé kombinace uvádíme minimální, maximální a průměrnou korelaci na jednotlivých testovacích sadách. Varianty jsou seřazeny sestupně podle dosažené průměrné korelace.

je fakt, že podle tabulky 4.1 korelují lépe méně četné sémantické slovní druhy. Ve vzorci CAP-MACRO však mají méně časté semposy větší váhu než v CAP-MICRO a celkové pokrytí tak koreluje lépe.

Protože aproximace APPROX-RESTR spolu s CAP-MACRO koreluje nejlépe, rozhodli jsme se, že v dalších experimentech budeme používat pouze tuto variantu. Dále v této práci budeme názvem SemPOS označovat právě tuto variantu.

Uvedené varianty metriky SemPOS jsme zkoumali také pro angličtinu. Výsledky těchto experimentů pro oba jazyky byly publikovány v článku [21].

5.5 Výsledky ostatních metrik

Kvalitu metrik uvedených v kapitole 3 jsme měřili stejně jako varianty metriky SemPOS. Výsledky naleznete v tabulce 5.3. Pod označením SemPOS se skrývá nejlépe korelující varianta z předchozí kapitoly[2]. Mezi zkoumané metriky jsme zařadili i lineární kombinaci metrik SemPOS a BLEU (v poměru 1:1) označovanou jako SemBLEU, protože jsme tuto metriku také použili pro optimalizaci vah.

Z tabulky je patrné, že nejlépe s lidmi koreluje metrika SemPOS. Po ní následuje lineární kombinace SemBLEU. Další metriky už ztrácejí na SemPOS více.

Zajímavé je, že metriky nezohledňující pořadí slov ve větě (SemPOS a PER) dopadly lépe než metriky, které pořadí slov zohledňují. Tento fakt by mohl být

[2]Korelace SemPOSu v tabulce 5.3 je mírně odlišná od stejné metriky v tabulce 5.2. To je způsobeno tím, že v tabulce variant SemPOSu jsme korelace nepočítali na sadě WMT11.

Metrika	Min.	Max.	Avg.
SemPOS	0.400	0.800	0.580
SemBLEU	0.090	0.800	0.524
PER	-0.090	0.800	0.410
CDER	0.140	0.560	0.370
BLEU	0.030	0.720	0.368
WER	-0.090	0.570	0.326
TER	-0.090	0.640	0.256

Tabulka 5.3: Metriky a jejich minimální, maximální a průměrná korelace. Metriky jsou seřazené sestupně podle dosažené korelace.

vysvětlen volným slovosledem v češtině: Anotátoři mohou označit překlad věty za kvalitní, i když má úplně jiné pořadí slov než referenční překlad.

Metrika CDER dopadla lépe než metrika WER a mohli bychom tedy konstatovat, že dovolení dlouhých skoků při počítání editační vzdálenosti pomohlo. To se na druhou stranu nedá říct o metrice TER, která naopak dopadla hůře než WER.

Také se potvrdilo, že metrika BLEU, která je považována za standardní metriku při automatické evaluaci překladu, nekoreluje v češtině tak dobře.

Povšimněte si také, že kromě nejlepší metriky, mají všechny metriky minimální korelaci velmi blízkou nule. Tyto nízké korelace vycházely většinou na testovacích sadách WMT08 a WMT09, kde jsme měli k dispozici výstupy pouze od čtyř systémů.

5.6 Metric Task

V sedmém ročníku WMT [10] jsme se zúčastnili s metrikou SemPOS soutěže Metric Task. V této soutěži ohodnotí každý účastník překlady zúčastněných překladových systémů pomocí své metriky. Z těchto hodnocení se vypočítá korelace s lidskými anotátory stejným způsobem jako v této práci.

Ve směru do angličtiny dosáhla metrika SemPOS průměrné korelace 0,9 a z dvanácti metrik vyhrála. Ve směru do češtiny dosáhla metrika korelace 0,52, což je srovnatelná hodnota s hodnotou v tabulce 5.3. Bohužel v tomto případě dosáhla většina metrik lepšího skóre a metriku SemPOS porazila.

6. Použití metrik v metodě MERT

V předchozí kapitole jsme zkoumali metriky z hlediska korelace s lidskými anotátory. V této kapitole se budeme věnovat vhodnosti metrik pro optimalizaci modelů v metodě MERT. Budeme tedy zkoumat, jak rychlý je výpočet metriky, jak rychle optimalizační algoritmus konverguje, zda má optimalizovaná funkce pěkný tvar atd. Pokud totiž použijeme při optimalizaci metriky, která nemá pěkný tvar, riskujeme, že nenalezneme globální minimum nebo že to bude trvat dlouho.

6.1 Provedené experimenty

Každou zkoumanou metriku jsme použili v metodě MERT pro optimalizaci anglicko-českého překladového systému. Tento systém je detailně popsán v článku [3] pod označením „baseline tF-F“, pro naši práci jsme získali již připravené modely. Jednalo se o jednoduchý frázový překlad slovních forem v systému Moses [17]. Překladový model byl natrénován na 197 tisících paralelních vět z korpusu CzEng 1.0 news [6]. Použitý jazykový model byl 5-gramový a byl natrénován na české straně korpusu CzEng 1.0 news.

Systém obsahoval dohromady 8 komponent (5 součástí překladového modelu, jazykový model, distortion model a word penalty) loglineárního modelu a bylo tedy potřeba nalézt 8 optimálních vah. Pro experimenty jsme použili implementaci metody MERT, která je také součástí nástroje Moses.

Jako trénovací korpus, na kterém metoda MERT hledala překlad nejlépe ohodnocený danou metrikou, jsme použili sadu 2489 vět z workshopu WMT10. Tuto sadu jsme měli v anglickém i českém překladu.

Protože běh metody MERT není deterministický (v Powellově algoritmu se začíná v náhodných bodech a hledá se v náhodných směrech), spustili jsme optimalizaci pro každou metriku pětkrát. Na konci každého běhu jsme získali nalezené váhy. Ty jsme pak použili pro překlad testovací sady vět (3003 vět z workshopu WMT11). Nakonec jsme překlad testovací sady ohodnotili všemi metrikami a výsledky jsme zprůměrovali přes použitou metriku při optimalizaci.

6.2 Výsledky

Výsledky provedených experimentů můžete nalézt v tabulce 6.1. V tabulce se nacházejí průměrná skóre různých metrik dosažená u modelů optimalizovaných na různé metriky a jejich směrodatné odchylky. V každém sloupci je tučně zvýrazněno nejlepší skóre. V tabulce 6.2 uvádíme průměrný počet iterací, průměrnou dobu běhu metody MERT a počet běhů, které dosáhly maximálního počtu iterací. V tabulce 6.3 uvádíme standardní odchylky jednotlivých vah, které vyšly z různých běhů optimalizace na danou metriku. Můžete tak vidět, které optimalizace jsou stabilní a dospějí vždy ke stejnému výsledku.

Protože jsme předpokládali, že samotná metrika SemPOS nebude vhodná pro optimalizování vah (je moc hrubá, protože nezohledňuje slovní formy a pořadí slov ve větě), zařadili jsme do výsledků i metriku SemBLEU, která počítá lineární kombinaci metrik SemPOS a BLEU v poměru 1:1.

Optimalizovaná metrika	Metrika použitá pro závěrečnou evaluaci						
	SemPOS	SemBLEU	PER	CDER	BLEU	WER	TER
SemPOS	**0.446 ± 0.011**	0.241 ± 0.008	−1.156 ± 1.507	0.231 ± 0.048	0.037 ± 0.021	0.116 ± 0.028	−1.252 ± 1.507
SemBLEU	0.393 ± 0.072	0.236 ± 0.057	0.140 ± 0.218	0.269 ± 0.087	0.079 ± 0.043	0.190 ± 0.074	0.023 ± 0.236
PER	0.371 ± 0.008	0.245 ± 0.006	**0.447 ± 0.001**	0.335 ± 0.007	0.119 ± 0.005	0.302 ± 0.008	0.306 ± 0.007
CDER	0.355 ± 0.004	0.238 ± 0.002	0.438 ± 0.002	**0.342 ± 0.000**	0.120 ± 0.001	**0.318 ± 0.001**	0.324 ± 0.001
BLEU	0.377 ± 0.002	**0.250 ± 0.001**	0.446 ± 0.001	0.339 ± 0.001	**0.123 ± 0.000**	0.308 ± 0.001	0.311 ± 0.001
WER	0.326 ± 0.006	0.219 ± 0.004	0.421 ± 0.002	0.337 ± 0.001	0.112 ± 0.001	0.317 ± 0.000	**0.325 ± 0.000**
TER	0.313 ± 0.008	0.211 ± 0.005	0.414 ± 0.005	0.333 ± 0.002	0.108 ± 0.002	0.316 ± 0.001	0.324 ± 0.001
SemBLEU*	0.425 ± 0.008	**0.261 ± 0.004**	0.234 ± 0.068	0.308 ± 0.005	0.098 ± 0.006	0.222 ± 0.017	0.126 ± 0.060

Tabulka 6.1: Tabulka výsledků optimalizace s různými metrikami. Řádek určuje metriku, která byla použita při optimalizaci. Sloupec určuje metriku použitou pro závěrečnou evaluaci optimalizovaného systému na testovací sadě. Protože byla metoda MERT spuštěna s každou metrikou pětkrát, nachází se v každé buňce průměrné skóre. V každém sloupci je tučně zvýrazněno nejlepší skóre. Metriky jsou seřazené podle dosažené korelace v předchozí podkapitole (viz. tabulka 5.3). V posledním řádku SemBLEU* se nacházejí hodnoty vypočítané z běhů MERTU optimalizovaných na metriku SemBLEU, které nedosáhly maximálního počtu iterací a stačily zkonvergovat.

Metrika	Průměrný počet iterací	Průměrná doba běhu	Počet dlouhých běhů
SemPOS	23.4 ± 3.6	62050 ± 14048 s	4
SemBLEU	15.0 ± 8.0	28311 ± 19073 s	1
SemBLEU*	12.5 ± 6.6	21635 ± 13708 s	0
PER	7.8 ± 2.5	2668 ± 966 s	0
CDER	8.4 ± 3.0	3376 ± 1489 s	0
BLEU	9.6 ± 3.0	4199 ± 2033 s	0
WER	6.0 ± 2.5	2018 ± 1032 s	0
TER	7.2 ± 1.9	4003 ± 1507 s	0

Tabulka 6.2: V této tabulce se nacházejí některé údaje z běhů metody MERT s různými metrikami. V prvních dvou sloupcích se nachází průměrný počet iterací a průměrná doba běhu. V posledním sloupci se nachází počet běhů (z celkového počtu 5), které skončily po dosažení maximálního počtu iterací.

Z tabulky je vidět, že optimalizace na metriku SemPOS dopadla velmi špatně. Skóre metriky SemPOS je sice nejlepší, ale hodnoty ostatních metrik jsou nejnižší. Navíc mají hodnoty některých metrik velmi vysokou standardní odchylku. To je způsobeno tím, že optimalizace na metriku SemPOS je velmi nestabilní a skončí pokaždé s odlišnými výsledky. To můžete vidět v tabulce 6.3, kde jsme pro každou optimalizovanou metriku spočítali standardní odchylky vah, které vyšly z různých běhů MERTu. Překlady testovacího korpusu přeložené systémem optimalizovaným na SemPOS byly na první pohled nekvalitní a překladač generoval velmi dlouhé věty.

Takto špatný výsledek souvisí nepřímo s tím, že optimalizace na metriku SemPOS ve čtyřech z pěti případů skončila, protože dosáhla maximálního počtu iterací (25). Akumulovaný n-best list v těchto případech nestačil zkonvergovat a předpovídané skóre se mohlo velmi lišit od skutečného. Optimalizátor tedy mohl v poslední iteraci nalézt váhy, které dávají podle n-best listu dobré výsledky, ale ve skutečnosti to dobré výsledky nejsou.

Optimalizace na lineární kombinaci SemBLEU dopadla o něco lépe, i když trpí podobnými nedostatky jako SemPOS. Ostatní metriky ohodnotily tuto optimalizaci také špatně. V neprospěch optimalizace na metriku SemBLEU hovoří i to, že některé jiné optimalizace dosáhly vyšší skóre metriky SemBLEU než tato optimalizace, která na SemBLEU optimalizovala. Protože výsledky této optimalizace pokazil jeden běh, který dosáhl maximálního počtu iterací, uvádíme pro úplnost v tabulkách 6.1 a 6.2 řádek SemBLEU*, který obsahuje výsledky zbývajících čtyř běhů, které nedosáhly maximálního počtu iterací.

Tabulka 6.2 navíc ukazuje, že optimalizace používající SemPOS a SemBLEU běžely řádově déle než ostatní metriky. To je způsobeno jednak větším počtem iterací a také tím, že metrika SemPOS potřebuje získat morfologické značky ve větě. (I když bylo značkování paralelizováno na 20 CPU, byla to nejvíce časově náročná část celé optimalizace.)

Ze všech těchto důvodů si myslíme, že metriky SemPOS a SemBLEU nejsou vhodné pro optimalizování modelů. Možné skutečné důvody, které stojí za tímto neúspěchem nabízíme dále v podsekci 6.2.1.

Optimalizace na metriky TER a WER už dopadly lépe. Skončily po menším

	Komponenty loglineárního modelu							
	d	**lm**	**w**	**tm1**	**tm2**	**tm3**	**tm4**	**tm5**
SemPOS	0.043	*0.072*	*0.183*	*0.055*	*0.056*	*0.091*	0.018	*0.362*
SemBLEU	*0.326*	0.053	0.127	*0.055*	0.035	**0.007**	*0.023*	0.316
PER	0.032	0.010	0.049	0.019	**0.007**	0.018	0.019	0.048
CDER	**0.014**	**0.005**	**0.029**	**0.011**	**0.007**	0.013	0.008	**0.020**
BLEU	0.027	0.014	0.043	0.016	0.019	**0.007**	0.015	0.087
WER	0.044	0.012	0.053	0.015	0.010	0.014	**0.007**	0.028
TER	0.020	0.013	0.072	0.029	0.015	0.023	0.018	0.054

Tabulka 6.3: Tabulka ukazuje standardní odchylky vah jednotlivých komponent loglineárního modelu, které byly získány z různých běhů MERTu s danou metrikou. V každém sloupci jsme zvýraznili nejvyšší standardní odchylku kurzivou a nejnižší tučně. Je tedy vidět, že nejlépe dopadla z tohoto pohledu metrika CDER a nejhůře metriky SemPOS a SemBLEU.

počtu iterací a hledaný vektor vah zkonvergoval vždy na velmi blízké hodnoty. Tyto optimalizace však nedosáhly nejvyšších skóre na stejných metrikách. Z tohoto důvodu a také z důvodu nejnižší korelace s lidskými anotátory si myslíme, že tyto metriky také nejsou vhodné pro optimalizaci.

Nejlépe dopadly dle našeho názoru metriky PER, CDER a BLEU. Všechny tři optimalizace konvergovaly k podobným vahám. Dosáhly vždy nejvyšších hodnot u metrik, na které se optimalizovalo, a u ostatních metrik se nejlepším výsledkům velmi blížily. Zvlášť dobře dopadla metrika PER, která má z této trojice nejvyšší korelaci s lidskými anotátory a navíc konverguje velmi rychle. Optimalizované váhy podle metriky CDER jsou zase podle tabulky 6.3 velmi blízko sebe.

6.2.1 Neúspěch metriky SemPOS

Důvodů pro neúspěch metriky SemPOS při optimalizaci bude pravděpodobně více. Jako nejpravděpodobnější vysvětlení se nám zdá to, že metrika SemPOS je velmi hrubá: mezi velmi podobnými větami neumí vybrat nejvhodnější překlad. Nezohledňuje totiž slovní formy, pomocná slova a pořadí slov, protože každou větu zredukuje na množinu t-lemmat plnovýznamových slov (V aproximaci approx-restr se navíc ponechávají t-lemmata pouze některých sémantických slovních druhů, a tím se rozlišovací schopnost metriky dále snižuje).

Tento problém se projevuje při hodnocení n-best listu, protože většina hypotéz se od sebe liší právě slovními formami a pořadím slov. Tudíž se při hodnocení vět z n-best listu mnohé věty zredukují na stejnou množinu a optimalizátor se pak nemá podle čeho rozhodovat, protože většina hypotéz má stejné skóre.

Abychom podpořili toto tvrzení, udělali jsme malý pokus: Vzali jsme n-best ($n = 100$) list vytvořený v první iteraci a spočítali jsme průměrný počet unikátních hypotéz na jednu vstupní větu. Těch je 37,94. To je na 100-best list relativně málo. Je to způsobeno tím, že dekodér občas vyprodukuje stejné věty různě (překlad věty se může lišit například segmentací) a s různým skóre. Pak jsme provedli redukci vět na množinu dvojic t-lemmat a semposů stejně, jako to dělá metrika SemPOS. Takových množin už bylo průměrně pouze 12,62 na jednu větu. V met-

rice SemPOS se dále tyto množiny použijí v průniku s referencí. Průměrný počet těchto množinových průniků, což je zároveň horní hranice počtu různých hodnot skóre metriky, na jednu větu je 3,02, což je opravdu velmi nízká hodnota.

Při optimalizaci se tento problém projevuje tak, že optimalizátor volí váhy téměř náhodně, protože má většina vět stejné skóre.

6.3 Tunable Metric Task

V rámci šestého ročníku WMT [12] proběhla soutěž Tunable Metric Task, jejímž cílem bylo použít vlastní metriku pro optimalizování statistického překladového systému z urdštiny do angličtiny. Optimalizované systémy byly použity pro překlad testovací sady vět. Tyto překlady pak byly vyhodnoceny lidskými anotátory.

Zúčastnili jsme se této soutěže s metrikou SemPOS. Použili jsme však kombinaci approx a cap-micro, protože nám tato varianta vycházela pro angličtinu lépe. Navíc jsme tuto metriku použili, ze stejných důvodů jako v této práci, v lineární kombinaci s metrikou BLEU v poměru 1:1.

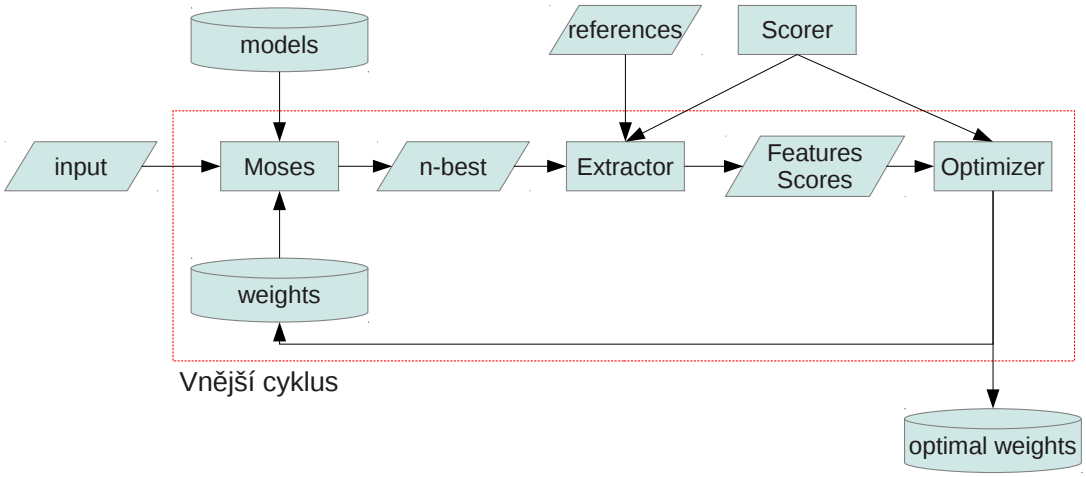
Na rozdíl od experimentů v češtině provedených v této práci jsme v soutěži Tunable Metric Task s metrikou SemBLEU nedopadli tak špatně. Optimalizované systémy byly hodnocené lidskými anotátory. V oficiálním hodnocení „$\geq$ others“ skončil náš systém pátý z osmi. V hodnocení „$>$ others“ náš systém dokonce zvítězil. Velký rozdíl v pořadí ukazuje, že na použitém způsobu výpočtu lidského skóre velmi záleží. Přitom však není jasné, který způsob je vhodnější. Pro úplnost ještě dodáváme, že v dalším způsobu výpočtu lidského hodnocení uvedeném v článku [20] dopadl náš systém také velmi dobře.

7. Implementace

7.1 MERT v překladači Moses

Pro experimenty s metodou MERT jsme využili implementaci popsanou v článku [1]. Tato implementace je součástí volně šiřitelného souboru nástrojů pro statistický překlad Moses [17]. Zdrojové kódy nástroje Moses jsou k dispozici v repozitáři na serveru Github[1]. Implementace MERTu se nachází v podadresáři `mert`.

Na obrázku 7.1 je znázorněn diagram této implementace. Vnější cyklus je napsán v jazyku perl ve skriptu `mert-moses.pl`, ze kterého se volají komponenty Extractor, Optimizer i samotný překladač Moses. Tyto komponenty jsou napsané v jazyku C++.



Obrázek 7.1: Diagram metody MERT implementované v Mosesu

Ve vnějším cyklu se přeloží trénovací korpus a vytvoří se n-best list obsahující n nejlepších překladů každé věty spolu s hodnotami charakteristických funkcí.

Program Extractor dostane na vstupu n-best list a vytvoří dva soubory statistik: Do souboru FeaturesFile uloží hodnoty charakteristických funkcí, které extrahuje z n-best listu. Do souboru ScoresFile pak pomocí dané metriky vypočítá statistiky metriky, ze kterých se později počítá hodnota metriky.

Program Optimizer dostane na vstupu soubory FeaturesFile a ScoresFile a počítá ve vnitřním cyklu Powellův algoritmus. Data ze souboru FeaturesFile se použijí pří hledání intervalů, na kterých se nemění vítězný překlad. Pro vypočtení hodnoty metriky daného výběru kandidátů z n-best listu se použijí statistiky ze souboru ScoresFile.

Nalezené optimální váhy se poté použijí pro překlad nového n-best listu, který je sloučen s předchozím n-best listem. Cyklus iteruje, dokud není splněna podmínka konvergence.

[1] `https://github.com/moses-smt/mosesdecoder`

7.2 Implementace metrik

Všechny metriky dědí ze základní abstraktní třídy `Scorer`. Tato třída definuje tři nejdůležitější funkce, které musí být implementovány v potomcích:

- `virtual void setReferenceFiles(...)` - V této funkci by měla metrika načíst referenční překlady a nějak si je předzpracovat. Například metrika BLEU si zakóduje a spočítá n-gramy.
- `virtual void prepareStats(...)` - Tato funkce připraví statistiky metriky pro danou větu. Například metriky WER, CDER, PER, ... si zde spočítají editační vzdálenost.
- `virtual void score(...)` - Tato funkce je volaná v programu Optimizer. Dostane indexy kandidátů a z připravených statistik vypočítá celkové skóre pro daný soubor kandidátů.

Protože většina metrik (zatím všechny metriky implementované v MERTu) sčítá statistiky přes všechny věty překladu a pak na ně aplikuje nějaký jednoduchý vzorec, existuje ještě jedna základní třída `StatisticsBasedScorer`, která dědí ze třídy `Scorer`. Tato třída implementuje metodu `score(...)`, ve které sečte odpovídající statistiky. Potomci této metody pak musejí implementovat metodu `virtual void calculateScore(...)`. To však většinou bývá pouze vypočítání hodnoty podle nějakého vzorce.

V rámci této práce jsme některé metriky přidali do implementace MERTu. V tabulce 7.1 naleznete seznam metrik aktuálně implementovaných v rámci sady pomocných nástrojů k překladači Moses. U každé metriky se nachází informace o tom, zda jsme v rámci této práce metriku navrhli nebo implementovali.

Metrika	**Vlastní návrh**	**Vlastní implementace**
SemPOS	ano	ano
PER	ne	ne
CDER	ne	ano
BLEU	ne	ne
WER	ne	ano
TER	ne	ne
InterpolatedScorer	ano	ano

Tabulka 7.1: Metriky implementované v MERTu

Kromě běžných metrik jsme také implementovali metriku InterpolatedScorer, která počítá lineární kombinaci libovolných metrik. Tuto metriku jsme použili pro výpočet SemBLEU.

Rozhodli jsme se použít implementované metriky i pro evaluaci. Do sady nástrojů překladače Moses jsme proto přidali program `evaluator`, který používá stejné třídy pro výpočet podobnosti mezi překladem a referencí.

8. Závěr

Tato práce zkoumala několik automatických metrik (SemPOS, SemBLEU, PER, WER, CDER, TER a BLEU) z hlediska korelace s lidskými anotátory v češtině a z hlediska použitelnosti těchto metrik při optimalizaci vah loglineárního modelu v metodě MERT.

Při této příležitosti jsme také navrhli některé aproximace a varianty metriky SemPOS. To se povedlo nad očekávání: metriku dokážeme počítat výrazně rychleji a nejlepší aproximace approx-restr+cap-micro koreluje s lidmi dokonce lépe než původní neaproximovaná metrika.

Nejlepší aproximace metriky SemPOS dopadla nejlépe z hlediska korelace s lidskými anotátory i ve srovnání s ostatními metrikami. Ukázalo se, že metriky nezohledňující pořadí slov ve větě (SemPOS a PER) korelují v češtině lépe než metriky, které pořadí slov zohledňují.

Při zkoumání metrik použitých při optimalizaci v metodě MERT se ukázalo, že metriky SemPOS a SemBLEU jsou pro optimalizaci nevhodné. Tyto metriky umějí dobře ohodnotit překlady od různých (již optimalizovaných) systémů, ale neporadí si s velmi podobnými větami z n-best listu. Pro účely optimalizace tedy navrhujeme používat metriky CDER a PER, které konvergují rychle a výpočet vah je stabilní. Nutno dodat, že i metrika BLEU, která se pro optimalizaci používá nejčastěji, nedopadla špatně.

8.1 Budoucí práce

Metrika SemPOS v této práci sice překonala ostatní metriky z hlediska korelace s lidskými anotátory, ale výsledky soutěže Metric Task [10] nabízejí moderní metriky, které jsou pro češtinu mnohem lepší než SemPOS. V další práci bychom proto měli tyto metriky implementovat a vyzkoušet při optimalizaci modelů.

Dále bychom měli zkoumat další možnosti a vylepšení metriky SemPOS. Z dosavadních pokusů vyplynulo, že některé sémantické slovní druhy vypovídají o kvalitě překladu méně a některé více. V dalších pokusech bychom proto chtěli vyzkoušet vážit pokrytí jednotlivých druhů při výpočtu celkového pokrytí. Také bychom rádi vyzkoušeli i jiné třídy než sémantické slovní druhy.

Pro lepší vyhodnocení systémů optimalizovaných na různé metriky bychom měli příště použít lidské hodnocení, stejně jako v soutěži Tunable Metric Task.

Literatura

[1] Nicola Bertoldi, Barry Haddow, and Jean-Baptiste Fouet. Improved minimum error rate training in moses. *The Prague Bulletin of Mathematical Linguistics*, 91:7–16, 2009.

[2] Ondřej Bojar, Miloš Ercegovčević, Martin Popel, and Omar Zaidan. A Grain of Salt for the WMT Manual Evaluation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 1–11, Edinburgh, Scotland, July 2011. Association for Computational Linguistics.

[3] Ondřej Bojar, Bushra Jawaid, and Amir Kamran. Probes in a Taxonomy of Factored Phrase-Based Models. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, Montreal, Canada, June 2012. Association for Computational Linguistics. Submitted.

[4] Ondrej Bojar and Kamil Kos. 2010 Failures in English-Czech Phrase-Based MT. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 60–66, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[5] Ondřej Bojar and Zdeněk Žabokrtský. CzEng0.9: Large Parallel Treebank with Rich Annotation. *Prague Bulletin of Mathematical Linguistics*, 92, 2009. in print.

[6] Ondřej Bojar, Zdeněk Žabokrtský, Ondřej Dušek, Petra Galuščáková, Martin Majliš, David Mareček, Jiří Maršík, Michal Novák, Martin Popel, and Aleš Tamchyna. The Joy of Parallelism with CzEng 1.0. In *Proceedings of LREC2012*, Istanbul, Turkey, May 2012. ELRA, European Language Resources Association. In print.

[7] Peter F. Brown, Vincent J.Della Pietra, Stephen A. Della Pietra, and Robert. L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19:263–311, 1993.

[8] Chris Callison-Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. Further meta-evaluation of machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 70–106, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[9] Chris Callison-Burch, Philipp Koehn, Christof Monz, Kay Peterson, Mark Przybocki, and Omar Zaidan. Findings of the 2010 joint workshop on statistical machine translation and metrics for machine translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 17–53, Uppsala, Sweden, July 2010. Association for Computational Linguistics. Revised August 2010.

[10] Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. Findings of the 2012 Workshop on Statistical Machine Translation. In *Proceedings of the Seventh Workshop on Statistical*

Machine Translation, Montreal, Canada, June 2012. Association for Computational Linguistics.

[11] Chris Callison-Burch, Philipp Koehn, Christof Monz, and Josh Schroeder. Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 1–28, Athens, Greece, March 2009. Association for Computational Linguistics.

[12] Chris Callison-Burch, Philipp Koehn, Christof Monz, and Omar Zaidan. Findings of the 2011 workshop on statistical machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 22–64, Edinburgh, Scotland, July 2011. Association for Computational Linguistics.

[13] Daniel Cer, Christopher D. Manning, and Daniel Jurafsky. The best lexical metric for phrase-based statistical mt system optimization. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 555–563, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[14] Michael Collins. Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. In *EMNLP '02: Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, pages 1–8, Morristown, NJ, USA, 2002. Association for Computational Linguistics.

[15] Jesús Giménez and Lluís Márquez. Linguistic Features for Automatic Evaluation of Heterogenous MT Systems. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 256–264, Prague, June 2007. Association for Computational Linguistics.

[16] Jesus Gimenez and Lluis Marquez. A smorgasbord of features for automatic MT evaluation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 195–198, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[17] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume, Proceedings of the Student Research Workshop, Proceedings of Demo and Poster Sessions, Tutorial Abstracts*, pages 177–180, Praha, Czechia, 2007. Association for Computational Linguistics.

[18] Kamil Kos and Ondřej Bojar. Evaluation of Machine Translation Metrics for Czech as the Target Language. *Prague Bulletin of Mathematical Linguistics*, 92, 2009.

[19] Gregor Leusch, Nicola Ueffing, and Hermann Ney. Cder: Efficient mt evaluation using block movements. In *In Proceedings of EACL*, pages 241–248, 2006.

[20] Adam Lopez. Putting Human Assessments of Machine Translation Systems in Order. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, Montreal, Canada, June 2012. Association for Computational Linguistics. Submitted.

[21] Matouš Macháček and Ondřej Bojar. Approximating a Deep-Syntactic Metric for MT Evaluation and Tuning. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 92–98, Edinburgh, Scotland, July 2011. Association for Computational Linguistics.

[22] Franz Josef Och. Minimum Error Rate Training in Statistical Machine Translation. In *Proc. of the Association for Computational Linguistics*, Sapporo, Japan, July 6-7 2003.

[23] Franz Josef Och and Hermann Ney. Discriminative training and maximum entropy models for statistical machine translation. pages 295–302, 2002.

[24] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. In *ACL 2002, Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, 2002.

[25] M. J. D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, 7(2):155–162, January 1964.

[26] Petr Sgall, Eva Hajičová, and Jarmila Panevová. *The Meaning of the Sentence and Its Semantic and Pragmatic Aspects*. Academia/Reidel Publishing Company, Prague, Czech Republic/Dordrecht, Netherlands, 1986.

[27] Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *In Proceedings of Association for Machine Translation in the Americas*, pages 223–231, 2006.

[28] Keh-Yih Su, Ming wen Wu, and Jing-Shin Chang. A new quantitative quality measure for machine translation systems, 1992.

[29] Warren Weaver. Translation. In William N. Locke and A. Donald Boothe, editors, *Machine Translation of Languages*, pages 15–23. MIT Press, 1949/1955. Reprinted from a memorandum written by Weaver in 1949.

[30] Zdeněk Žabokrtský, Jan Ptáček, and Petr Pajas. TectoMT: Highly modular MT system with tectogrammatics used as transfer layer. In *ACL 2008 WMT: Proceedings of the Third Workshop on Statistical Machine Translation*, pages 167–170, Columbus, OH, USA, 2008. Association for Computational Linguistics.

Seznam tabulek