Pokročilé metody rozpoznávání řeči

Přednáška 9

Rozpoznávání s velkými slovníky,
pravděpodobnostní LM,
nové a alternativní přístupy k ASR

Poznámky k předchozí úloze

Úloha:

Rozpoznání 173 vět za velmi specifických podmínek – slovník sestavený pouze ze slov v těchto větách a s 2 typy LM

a) **Zerogram (všechna slova mohou po sobě následovat se stejnou pravděpodobností)**
   - **Acc mezi 67 – 78 % (vliv zejména výslovností)**

b) **Bigram (vytvořený na testovacích větách)**
   - **Acc mezi 90 – 100 %**

Pozn. Testovací data opravena 9.11.2023

Rozpoznávání s velkým slovníkem (1)

Úlohy zaměřené na diktování či přepis řeči vyžadují velké slovníky

Kolik slov je třeba?

Angličtina versus čeština (AJ 50 tis. slov, ČJ > 500 tis. slov)

Angličtina	**Čeština**
driver, drivers, driver's	řidič, řidiče, řidiči, řidičem, …. řidička, řidičův, řidiččin, řidičský,
important	důležitý, důležitého, důležitá, … důležitější, nejdůležitější, ….
build, builds, built, building	stavit, stavím, stavíš, stavili, stavily, stav, stavící, vystavit, postavit, zastavit, nestavit

Rozpoznávání s velkým slovníkem (2)

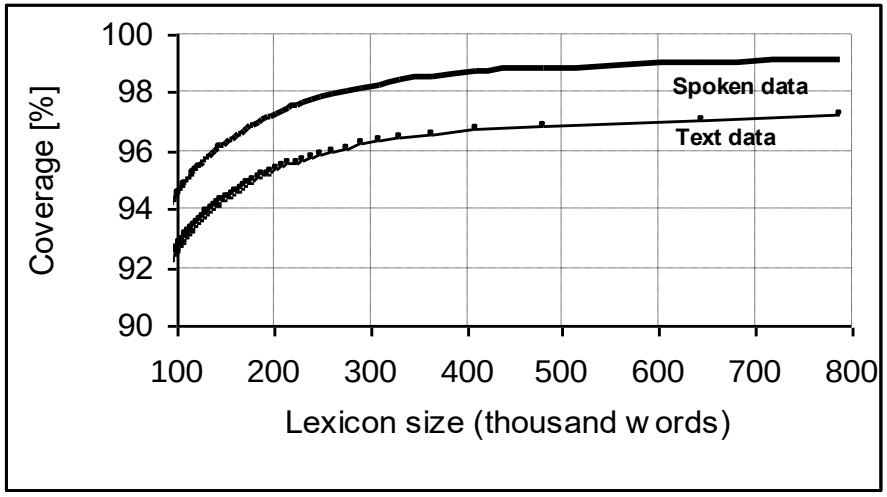
Jak vytvořit vhodný slovník s přijatelnou velikostí?

1. Získat **co nejvíce textů** z dané oblasti (ideální zdroj Internet)
   - všeobecný slovník: směs textů novinových, odborných, beletrie …
   - odborný slovník: texty pouze z dané oblasti
2. Texty je třeba **vyčistit a znormalizovat**
   - odstranit netextové položky (obrázky, znakovou grafiku, formátovací symboly, atd.)
   - identifikovat a odstranit časté **překlepy** (např. pomocí spell-checkeru, není jednoduché)
   - **rozvinout číslovky** psané číslicí (nejednoznačná úloha) a podobně též **zkratky** (které se nevyslovují jako zkratky, např. „tzv.“, „km/hod“)
3. Identifikovat slova, která mají různé **ortografické podoby** (např. „milion“ a „milión“, nebo „president“ a „prezident“) a **sjednotit je** do jediné vybrané (četnější) podoby

Rozpoznávání s velkým slovníkem (3)

Jak vytvořit vhodný slovník s přijatelnou velikostí? (pokračování)

3. **Sestavit seznam všech slov** vyskytujících se v daných textech a seřadit jej podle četnosti
4. Do slovníku **vybrat prvních N slov podle četnosti**
   - N se volí podle možností/omezení rozpoznávacího systému (např. 60k u HVite) nebo na základě požadovaného pokrytí (coverage rate)



Poznámka: Slova mimo slovník (OOV – Out-of-Vocabulary) nebudou nikdy rozpoznána
OOV rate = 100 – Coverage rate [%]

Rozpoznávání s velkým slovníkem (4)

Vliv velikosti slovníku na OOV, ACC (přepis zpráv, 2005)

Slovník	Počet slov	Min. frekvence	Accuracy [%]	OOV [%]
64K	64620	300	70,96	5,17
102K	102228	140	73,75	3,31
149K	148928	70	75,62	1,94
195K	194942	40	76,64	1,34
257K	257056	20	77,27	0,97
312K	312289	10	78,13	0,75

Rozpoznávání s velkým slovníkem (5)

Jak vytvořit vhodný slovník s přijatelnou velikostí? (pokračování)

6. Ke každému slovu dodat **výslovnost.** (Speciality: zkratky, cizí jména).
7. U některých (obvykle nejčastějších) slov přidat **výslovnostní varianty** (např. „v – v, f“, „nad – nat, nad“, „CD – cédé, sídí“, „Klause – klause, klauze“)
8. Pro další (např. sémantické) zpracování může slovník obsahovat **další přídavné informace:**

Standardní ortografie	**Alternativní ortografie**	**Výslovnostní varianty**	**Morfologická třída**	**Základní tvar (lemma)**
milionů	miliónů	milijonú, milijónú	Num2P Num4P	milion
téze	these, teze	téze, teze	NounMasc1P	téze
s		s, z, sE, zE	Prep4, Prep7	s

Pravděpodobnostní jazykový model

Běžná řeč se neřídí pevnými pravidly. Možná je prakticky každá kombinace slov, liší se však svou pravděpodobností.

Možné jsou i zdánlivě nepřijatelné kombinace, např. „krásná mladík“

„Byla krásná, mladík se za ní ohlédl.“

Výhoda pravděpodobnostního LM: lze ho kombinovat s rovněž pravděpodobnostním AM, jen je nutné je vzájemně vybalancovat. To umožňuje přepínač -s u HVite

Tvůrce konceptu pravděpodobnostního LM:

Bedřich (Frederick) Jelínek (1932-2010, Johns Hopkins Uni., IBM)

Nástroje pro vytvoření LM

HTK v základní verzi podporuje práci s bigramy

Nástroje:

- HLStats, HBuild

Popis:

- HTKBook, zejména kapitola 12

SRILM (SRI Language Modeling Toolkit)

pravděpodobně nejvíce používaný nástroj pro tvorbu LM, jeho výstupy lze použít pro rozpoznávání v rámci HTK

http://www.speech.sri.com/projects/srilm/

Možnosti a omezení HTK

- Lze v něm vyvinout rozpoznávač použitelný v řadě aplikačních oblastí
- Možné je i využití v komerčních aplikacích.
- U HVite omezení slovníku na cca 60k slov.
- Příliš pomalé rozpoznávání u větších slovníků
- HDecode – rychlejší rozpoznávač (oproti HVite), AM musí být v podobě trifonů (typu cross-word), možné i trigramy.
- Vývoj v letech 1990-2010, nezachytil už nástup neuronových sítí.
- Trik jak rozpoznávat češtinu s diakritikou:
  vytvořit si oboustranný převodník typu:

  může <–> muze1

  muže <–> muze2

  …

  Pro přípravu, rozpoznávání a vyhodnocování použít slovník a LM bez diakritiky, výstup převést na český slovník

Alternativy k HTK

Systém Julius

- vyvinutý v Japonsku v letech 1997-2005
- open source, trénování vychází z HTK
- umožňuje rychlý vývoj systémů pracujících v reálném čase (i s velkým slovníkem)
- **http://julius.osdn.jp/en_index.php**

Systém Kaldi:

- v období 2010-2018 nejlepší open source nástroj pro ASR
- podporuje práci s neuronovými sítěmi
- na vývoji se podílel i Jan Silovský z TUL
- **http://kaldi-asr.org/**

Neuronové sítě v ASR (1)

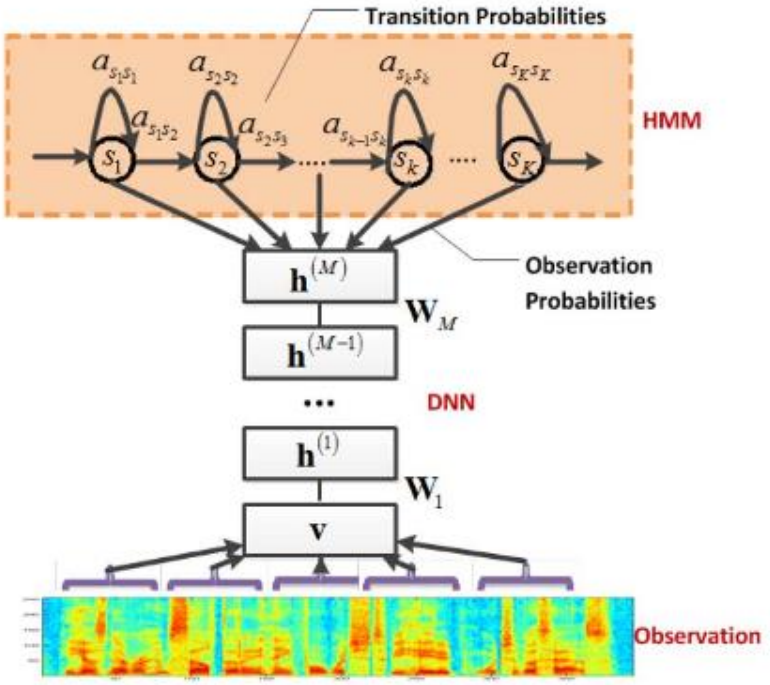
V **akustickém modelu** nahrazují gaussovské směsi (GMM), HMM zůstává

-> tzv. **hybridní model** – DNN-HMM

architektury FDNN (dopředná), CNN (konvoluční), RNN (rekurentní)

Výhody NN:

- lepší schopnost modelovat rozložení příznaků v prostoru
- trénování je **diskriminativní** (parametry pro všechny hlásky se učí současně tak, aby v daném framu zvítězila správná hláska)



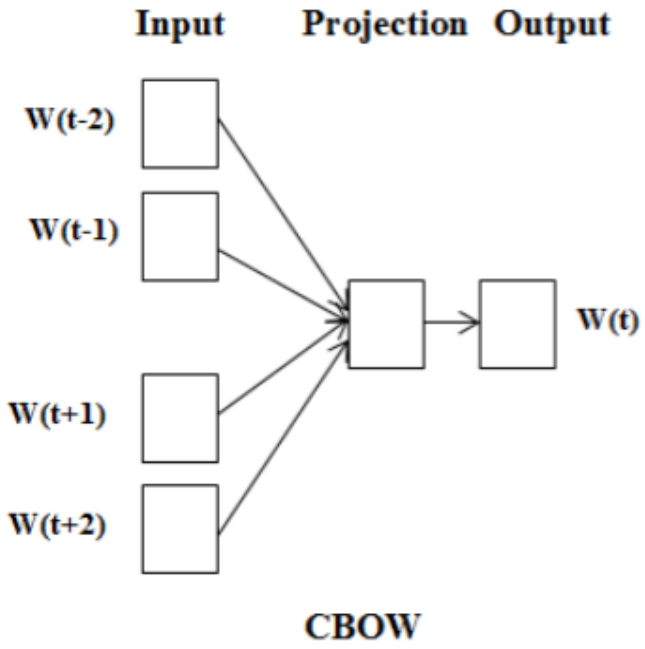
Neuronové sítě v ASR (2)

Neuronové sítě se prosazují i v **jazykovém modelování**.

Koncept **Word to Vector** - snaha reprezentovat slova číselnými vektory, autor Tomáš Mikolov a kol. (2013)

Předchozí snahy reprezentovat slova jako čísla:

- skalár (index slova ve slovníku)
- one-hot vektor [0, 0, 0, .., 1, 0, …0]



Princip word2vec:

Dvouvrstvá síť se učí predikovat slova na základě kontextu ve větě, vektory naučených vah jsou pak reprezentací jednotlivých slov
Dimenze vektorů jsou rel. malé (~100)

Slova semanticky podobná leží poblíž sebe

Neuronové sítě v ASR (3)

Nejnovější přístup: **End-to-end** (E2E)

Princip: poměrně složitá (hluboká rekurentní) neuronová síť se učí převádět signál (reprezentovaný příznaky) **přímo na sekvenci písmen**.

Výhody:

- není třeba sestavovat slovník (a zejména výslovnosti),
- „slovník“ tvoří seznam znaků (písmena + mezera + speciální symbol „blank“)
- síť se učí přímo s konečným cílem správně přepisovat text
- neučí se zvlášť akustický a jazykový model (a tudíž není třeba je balancovat)
- při dostatečně velkém množství trénovacích dat dává lepší výsledky

Nevýhody:

- vyžaduje mnohem větší (10x - 100x) množství nahrávek s přepisy
- není snadné doučit síť, aby správně přepisovala i neviděná slova (zejména s nepravidelnou výslovností, s méně obvyklými znaky, atd.)

Příklad systému E2E (1)

Trénovací strategie CTC (Connectionist Temporal Classification)

Princip: strategie pro neuronové sítě, které se mají naučit převod mezi různě dlouhými sekvencemi různých typů, např. převod

- sekvence framů (příznak. vektorů) na sekvenci fonémů nebo písmen,
- sekvence řezů v obrázku na sekvenci písmen ručně psaného textu,
- sekvence znaků/slov v jednom jazyce na sekvenci znaků/slov v 2. jazyce

Základní použití v ASR:

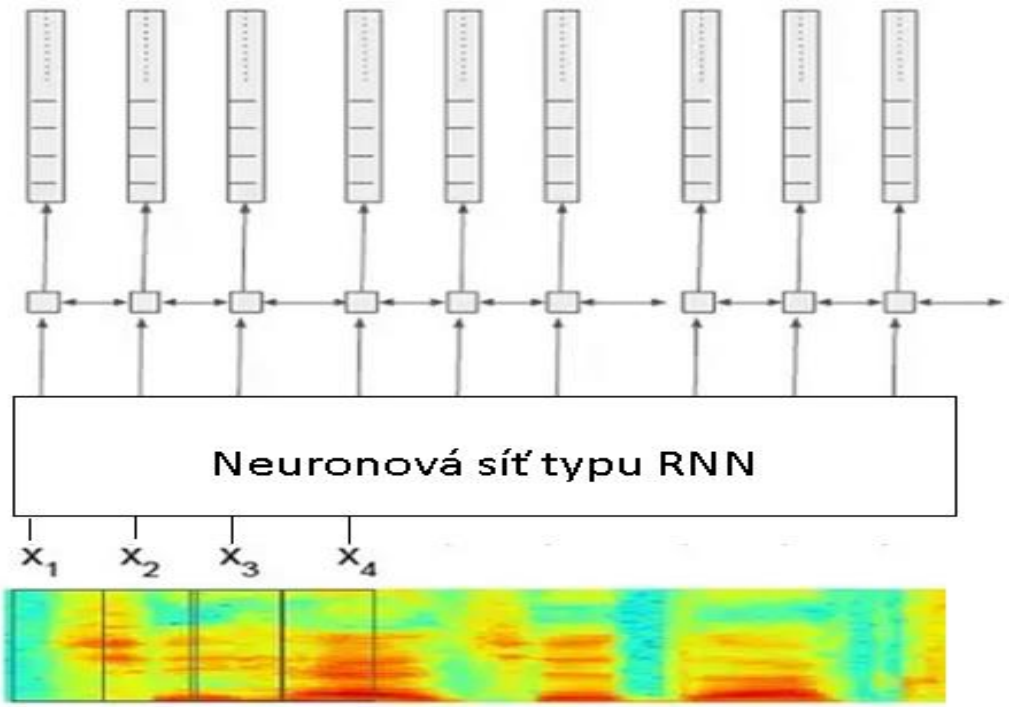
vstupem do neuronové sítě jsou příznakové vektory všech framů nahrávky

mezivýstupem (pro každý frame) jsou pravděpodobnosti písmen daného jazyka (včetně mezery a znaku „blank“ – symbol <b> nebo **ε** , celkem **L** znaků)

finálním výstupem je pak nejpravděpodobnější sekvence znaků (písmen a mezer) tvořící text, finální výstup vytváří dekodér

Příklad systému E2E (2)

mezivýstup - výstup neuronové sítě pro jednotlivé framy



pro každý frame a každý znak získáme log. pravděpodobnosti

výstupní vektor projde funkcí softmax

RNN produkuje pro každý frame vektor výstupů (o dimenzi L)

příznakové vektory pro framy

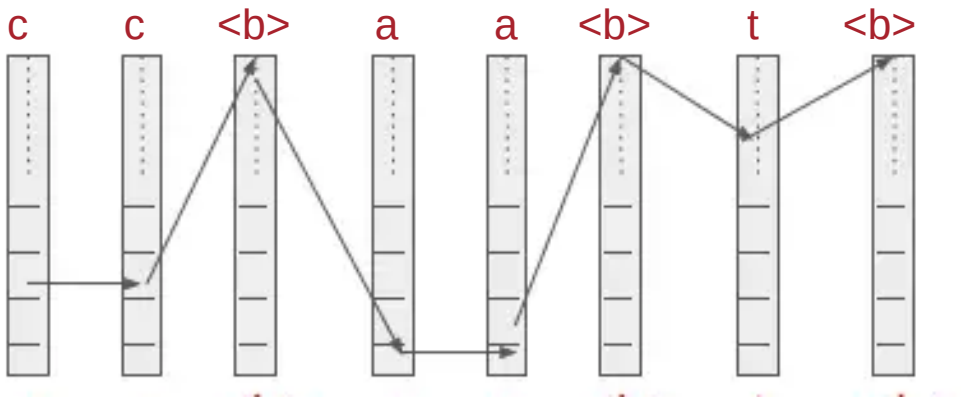
řečový signál

vstup

„Slovník“ systému má velikost L a tvoří ho povolené znaky jazyka (např. *a, b, c, d, ……y, z, [space], <b>*

Příklad systému E2E (3)

Příklad jednoduché dekódovací strategie (Greedy decoding)



znaky s nejvyšším skóre

Postup:

1) V každém framu najdi znak s nejvyšším skóre (log. pravděpodobností)
2) Sestav z těchto znaků sekvenci: cc<b>aaaa<b>t<b>
3) Za sebou opakující se znaky nahraď jediným: c<b>a<b>t<b>
4) Vymaž všechny znaky <b> - př. cat

Pozn. Znak <b> je nutný pro správné dekódování slov typu „oddych“, „zoo“, „měkký“ …

Varianty E2E systémů

1) **Lepší dekódovací strategie:**
   Prefix beam search (obdoba Viterbiho algoritmu známého z HMM)

   Princip: ke slovu cat z předchozího příkladu můžeme dojít z více sekvencí, např.
   cc<b>aa<b>t<b>, c<b>aaa<b>t<b>, <b>caatt <b> , ccaaaaat
   Strategie tedy hledá globálně nejlepší sekvenci (místo lokálně nejlepší) a je založená na dynamickém programování.

   Podrobné vysvětlení např. zde: https://distill.pub/2017/ctc/#inference

   kód v Pythonu: https://github.com/HawkAaron/E2E-ASR/blob/master/ctc_decoder.py

2) **Lepší „slovník“:**
   Substringy (znaky + nejčastěji se vyskytující slovní podřetězce, 1000 - 5000)

3) **Sítě typu Transformer, Enkodér – Dekodér, …**

4) **Kombinace Prefix beam search a N-gramového modelu**

Dnešní úloha

Sestavte si CTC dekodér typu Greedy search a vyzkoušejte ho na testovacích datech z minula.

1. Stáhněte si z e-learningu soubor data-E2E.TAR. Uvnitř najdete data.zip, kde jsou pro každou testovací nahrávku uloženy mezivýstupy z RNN sítě natrénované na cca 500 hodinách české řeči (zejména záznamy z parlamentu). Hodnoty jsou v matici o rozměru F x L (počet framů x počet znaků ve „slovníku“).
2. Seznam znaků (s jejich indexy) najdete v souboru znaky_pmr.txt, z něhož je také vidět, jak výše uvedené soubory načíst v Pythonu.
3. Dekodér nejsnáze napíšete v Pythonu, ale můžete i v jiném jazyce.
4. Volitelně si můžete vyzkoušet i dekodér typu Prefix Beam Search a porovnat.
5. Váš dekodér, výstupy z něj pro každý soubor a celkovou úspěšnost testu mi pošlete do konce tohoto týdne.

Pozn.1 Pro vyhodnocení úspěšnosti využijte program HResults, pro nějž vytvoříte standardní soubor recout.mlf. Můžete očekávat WAcc kolem 77 %.

Závěrečná úloha

Vytvořte systém rozpoznávání spojité řeči pracující v prostředí HTK

1. Stáhněte si z e-learningu malý korpus českých vět (~10 000 vět z přepisů rozhlasu, ~500 000 slov, ~50 000 různých).
2. Zpracujte textový soubor tak, abyste dostali seznam slov podle četnosti.
3. Ze všech slov vytvořte z něj master-slovník a ke každému automaticky doplňte výslovnosti. (U slov z minulé úlohy můžete převzít již hotové výslovnosti.)
4. Následně převeďte master-slovník i korpus na malá písmena a bez diakritiky.
5. Z master-slovníku vytvořte slovník s N nejčastějšími slovy a pomocí nástrojů HTK nebo SRILM vytvořte příslušný bigramový LM.
6. Otestujte je na test setu (173 vět). Experimenty proveďte pro N = 10k, 20k a 50k (master-slovník). Kromě Accuracy a Correctness zjistěte také OOV rate. (V rámci experimentování najděte nejvhodnější hodnoty pro parametry –s a –p a podívejte se rovněž na význam –t, který pomůže v urychlení).
7. Zdokumentuje výsledky a spolu s vašimi programy/skripty zašlete vyučujícímu, a to do 31.12.2023.