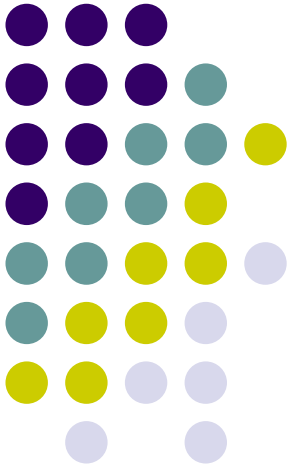
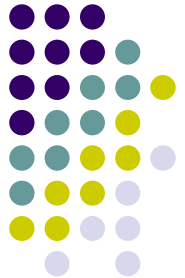
Rozpoznávání písmen

Jiří Šejnoha

Rudolf Kadlec

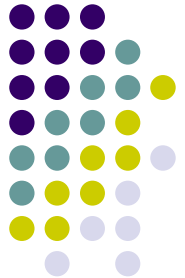


Osnova



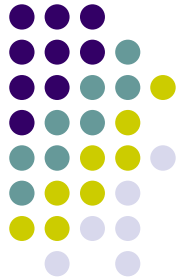
- Motivace
- Popis problému
- Povaha dat
- Neuronová síť
  - Architektura
  - Výsledky
- Zhodnocení a závěr

Popis problému



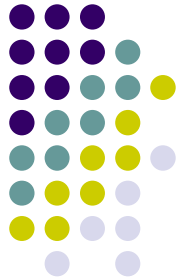
- Jedná se o praktický problém, kdy je potřebné z grafické předlohy extrahovat textová data a jejich význam resp. jejich základní detekce a kategorizace
- Zaměříme se pouze na část problematiky a to na roztřídění z předem detekovaných a statisticky předzpracovaných písmen textu

Popis problému 2.



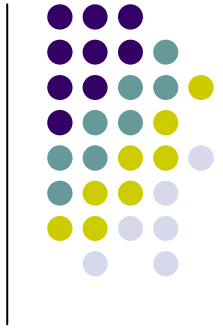
- Naším cílem je tedy z předem zpracovaných dat – grafické a statistické charakteristiky písmen abecedy, detekovat o jaké písmeno abecedy se jedná

Zdrojová data



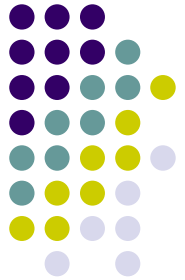
- 20 tisíc vzorků písmen, písmena jsou psána různými fonty a je k nim přidán náhodný šum
- Přibližně stejná distribuce všech písmen latinské abecedy, od každého cca 700 vzorků
- Vzorek je dvojice (charakteristika,písmeno)
- Charakteristika je 16 údajů získaných z grafické podoby písmene

Charakteristiky



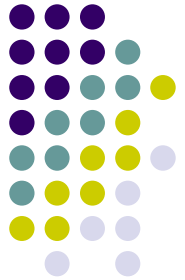
- Převážně statistické údaje
  - x-bar - mean x of on pixels in box (integer)
  - y-bar - mean y of on pixels in box (integer)
  - x2bar - mean x variance (integer)
  - y2bar - mean y variance (integer)
  - xybar - mean x y correlation (integer)
  - x2ybr - mean of x * x * y (integer)
  - xy2br - mean of x * y * y (integer)
  - x-ege - mean edge count left to right (integer)
  - …
- Příklad jedné charakteristiky písmene
  - K 3 6 5 4 3 3 9 2 6 10 11 11 3 8 2 6
    K 7 11 10 8 6 4 8 3 7 11 10 12 4 8 4 6

Neuronová síť



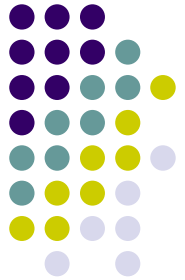
- Zvolili jsme *dopřednou vrstevnatou* neuronovou síť, trénovanou algoritmem *backpropagation*
- Původní záměr byl rozpoznávat všech 26 písmen - přílišné časové a paměťové nároky pro učení…
- Nakonec jen prvních 13 písmen – přijatelnější nároky
- Na simulaci a výpočet neuronové sítě byl použit toolbox na práci s neuronovými sítěmi programu MATLAB, síť byla zadána pomocí skriptu a distribuována na více počítačů

Architektura



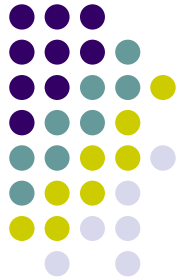
- Vstup 16 charakteristik => 16 neuronů
- Skrytá vrstva
  - Počet neuronů počítán přes for cyklus - zkoušeno od 2 do 28
  - Vyšší počet nebyl možný vzhledem k paměťovým nárokům procesu
- Výstup je písmeno A-L transformované do posloupnosti 0(neaktivní neuron) a 1(aktivní, tj. výstup definován > 0.8), kde muže být jen jedna 1, pořadí 1 určuje písmeno => 13 neuronů
- Přenosové funkce *logsig, logsig*

Metodika trénování



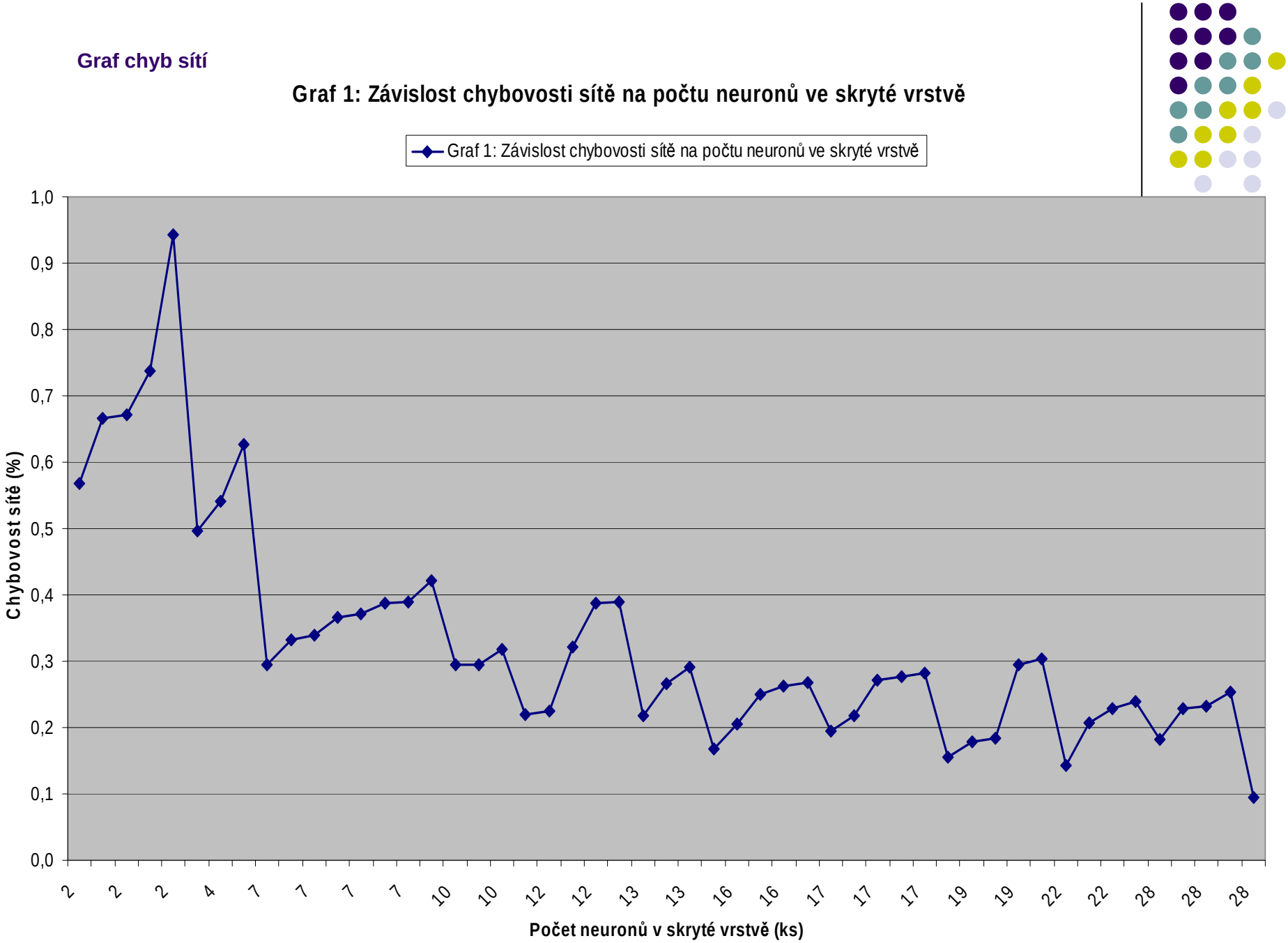
- Množinu vzorů tvořilo prvních 13 písmen abecedy po 700 ks na písmeno. Ta byla zpermutována a 65% dat bylo použito jako trénovací a 35 % jako validační množina
- Při procesu učení byla použita vnitřní funkce *train* s podmínkou ukončení trénování při vzrůstu chyby na validační množině
  - Odzkoušením bylo zjištěno, že použití této funkce nemá větší vliv na schopnost natrénování sítě a vyhnuli se tak problému přeučení sítě a navíc došlo k výraznému urychlení procesu učení

Metodika trénování 2.

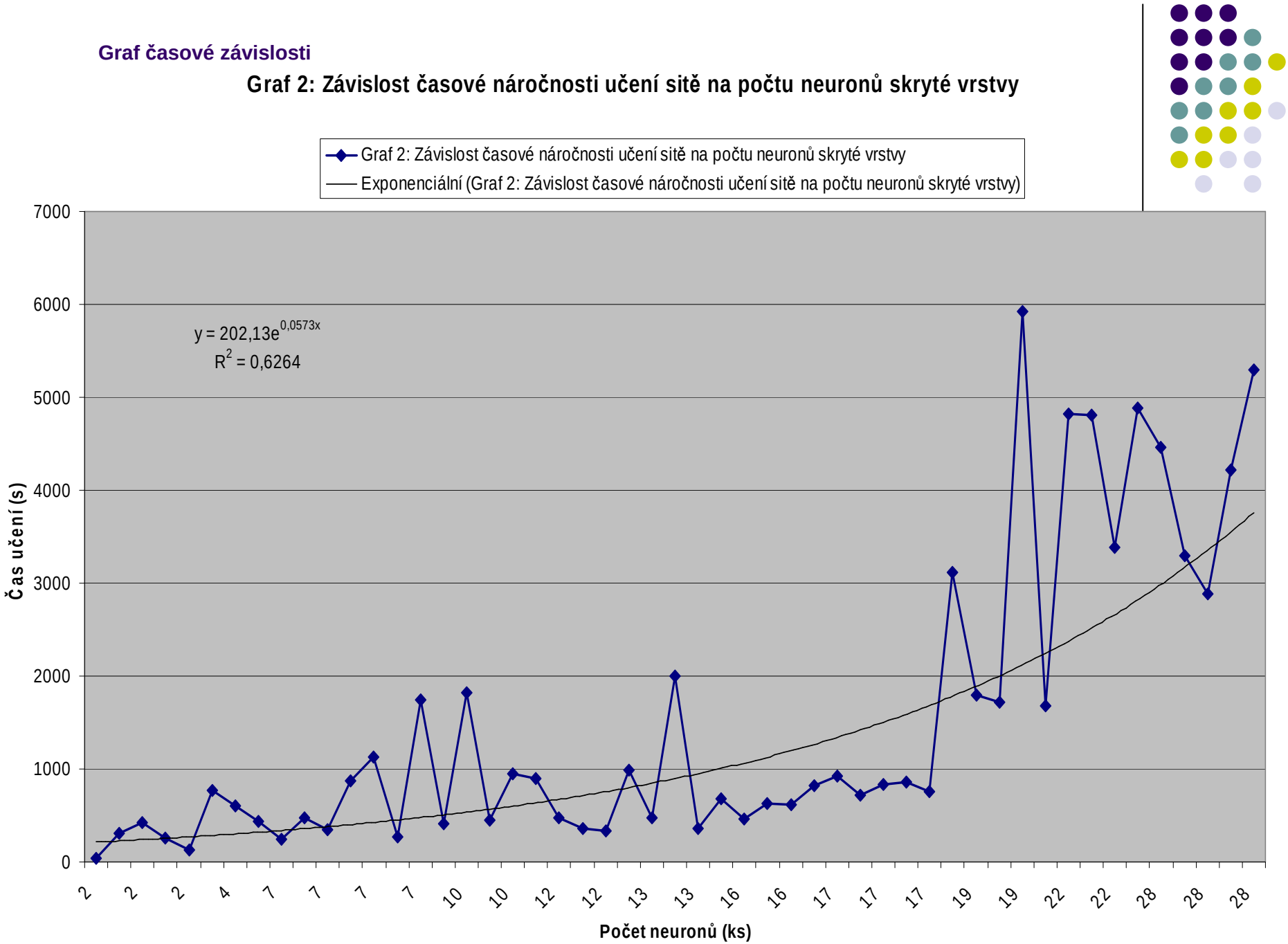


- Pro každou architekturu sítě (tj.: pro každý počet neuronů ve skryté vrstvě) byla síť učena 5x a pokaždé náhodně inicializována
- Rozdíl v naučení pro různé počáteční nastavení byl až v desítkách procent
- Rozdíly časů byly až násobné
- Větším počtem inicializací jsme omezili uvíznutí ve „špatném“ lokálním minimu

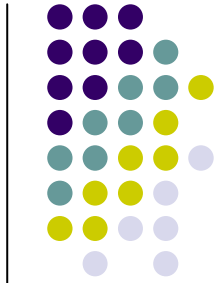
Graf chyb sítí



Graf časové závislosti

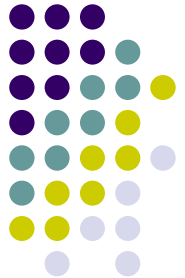


Shrnutí výsledků



- *Při nejlepším naučení sítě bylo dosaženo **9,5%** chyby a to při konfiguraci sítě **16-28-13**, tedy s 28 neurony v skryté vrstvě*
- Při zkoumání velikosti chyby sítě v závislosti na jejím počtu neuronů lze pozorovat, že chyba sítě klesá, ne však lineárně, ale pravděpodobně(dle grafu 1) exponencielně
- Lze předpokládat, že při velkém počtu neuronů, by síť menší chyby nedosahovala, pouze by se „přeučila“

Pár poznámek na konec



- „Větší“ sítě nebylo možno natrénovat vzhledem k výpočetním možnostem
- Čistý čas výpočtů 21 hodin, hrubý(testování, pády na kapacity pamětí a CPU Time,…) několik dní