Rede neural para classificação de números manuscritos

Rafael Hengen Ribeiro, Regis Thiago Feyh, Ricardo Parizotto

4 de Julho de 2016

Conteúdo

1 Dependências **2**
- 1.1 Biblioteca FANN (*Fast Artificial Neural Network*) . . . 2
  - 1.1.1 Ubuntu/Linux Mint/Debian . . . 2
  - 1.1.2 Fedora 21-22/Red Hat/CentOS . . . 2
  - 1.1.3 Fedora 23-24 . . . 2
- 1.2 (Opcional) *Plotly* - Geração de gráficos com *python* . . . 2
  - 1.2.1 Instalação do *pip* . . . 2
  - 1.2.2 Instalação do Plotly . . . 2

2 Utilização do programa **3**
- 2.1 Compilação . . . 3
- 2.2 Treinamento . . . 3
- 2.3 Teste . . . 4

3 Resultados obtidos **5**

Capítulo 1

Dependências

1.1 Biblioteca FANN (*Fast Artificial Neural Network*)

Para a compilação e utilização do programa é necessário ter a biblioteca FANN instalada. Segue abaixo um tutorial de instalação em várias distribuições *linux*.

1.1.1 Ubuntu/Linux Mint/Debian

```
sudo apt-get install libfann{2,-dev}
```

1.1.2 Fedora 21-22/Red Hat/CentOS

```
su -c "yum install fann fann-devel"
```

1.1.3 Fedora 23-24

```
sudo dnf install fann fann-devel
```

1.2 (Opcional) *Plotly* - Geração de gráficos com *python*

Para a geração de gráficos é necessário ter um interpretador *python* com a versão >= 2.7, o programa *pip* e a biblioteca *plotly*, cuja instalação é descrita abaixo.

1.2.1 Instalação do *pip*

Fedora 21-22/Red Hat/CentOS

Python 2.7: `su -c "yum install python-pip"`
Python 3: `su -c "yum install python3-pip"`

Fedora 23-24

Python 2.7: `sudo dnf install python-pip`
Python 3: `sudo dnf install python3-pip`

Ubuntu/Linux Mint/Debian

Python 2.7: `sudo apt-get install python-pip`
Python 3: `sudo apt-get install python3-pip`

1.2.2 Instalação do Plotly

Python 2.7: `sudo pip install plotly`
Python 3: `sudo pip3 install plotly`

Capítulo 2

Utilização do programa

2.1 Compilação

Para compilar o programa, após a instalação de todas as dependências listadas no capítulo 1, acesse a pasta raiz do programa através do terminal e execute o comando: `make`.

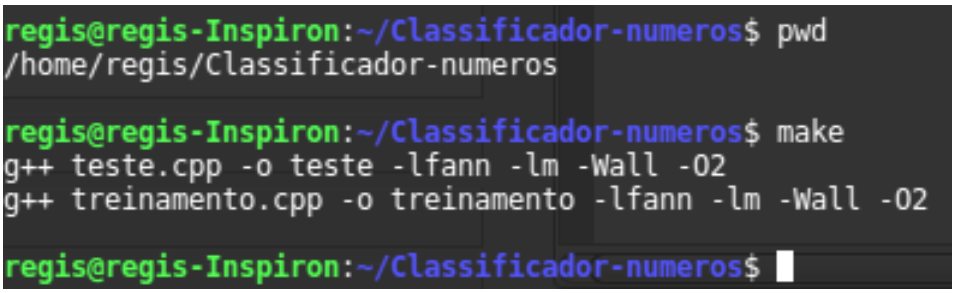

Figura 2.1: Exemplo de compilação

2.2 Treinamento

O treinamento da rede neural é feito a partir do programa `treinamento`. Para treinar a rede deve-se executar `./treinamento` < *qtd_neuronios* >, onde *qtd_neuronios* é a quantidade de neurônios que terá na camada escondida. O arquivo para o treinamento está disponível em `input/treinamento` com o nome `optdigits2.tra`.

Durante a execução do programa são mostrados o número da época atual, o erro médio ao quadrado, doravante chamado de MSE (*Mean Squared Error*), e a porcentagem de erro. Na figura 2.2 é mostrada uma execução parcial do programa.

Ao final da execução o programa salva a configuração da rede em um arquivo disponível no diretório `melhores/` com o nome `number_classify.best_h`<*n*>`.net`, onde *n* é o número de neurônios na camada escondida. Além deste arquivo, são salvos os dados da execução para estatísticas no arquivo `estatisticas/hidden`<*n*>`.txt`

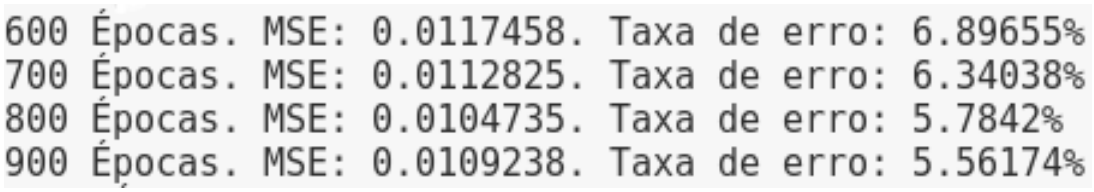

Figura 2.2: Exemplo de execução do programa para 20 neurônios

2.3 Teste

O programa `teste` lê o arquivo com a extensão `.net`, que é passado por parâmetro, gerado pelo treinamento, o qual possui a rede que obteve o menor erro durante a fase de treinamento. O programa `teste` executa a rede neural com uma base diferente da utilizada no treinamento e validação, no intuito de verificar a taxa de acerto.

Capítulo 3

Resultados obtidos

Foram treinadas redes variando o número de neurônios escondidos entre 10 e 600. No gráfico da imagem 3.1 é possível visualizar a variação da taxa de acerto:

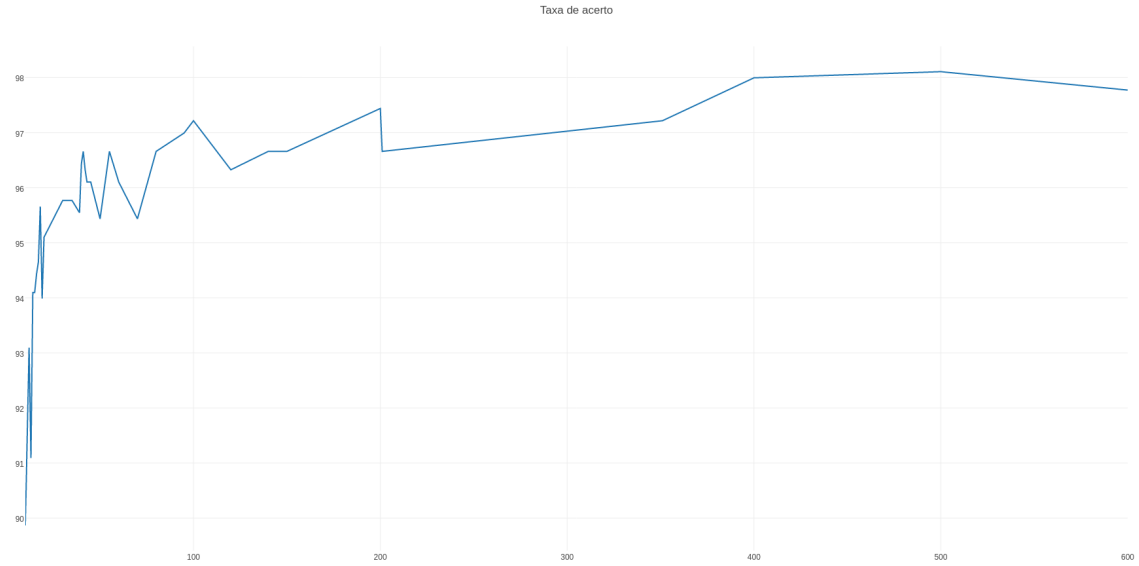

Figura 3.1: Variação da taxa de acerto conforme o número de neurônios escondidos

A taxa de acerto máxima foi observada com 500 neurônios escondidos, acertando **98.1069%**. No entanto, é possível observar 97.216% já com 100 neurônios, e 96.6592% com 41 neurônios escondidos. Com 10 neurônios escondidos foi obtida uma taxa de acerto de 89.8664%.

Nas figuras a seguir é possível visualizar a variação do *Mean Squared Error* ao longo das épocas, variando-se o número de neurônios escondidos.

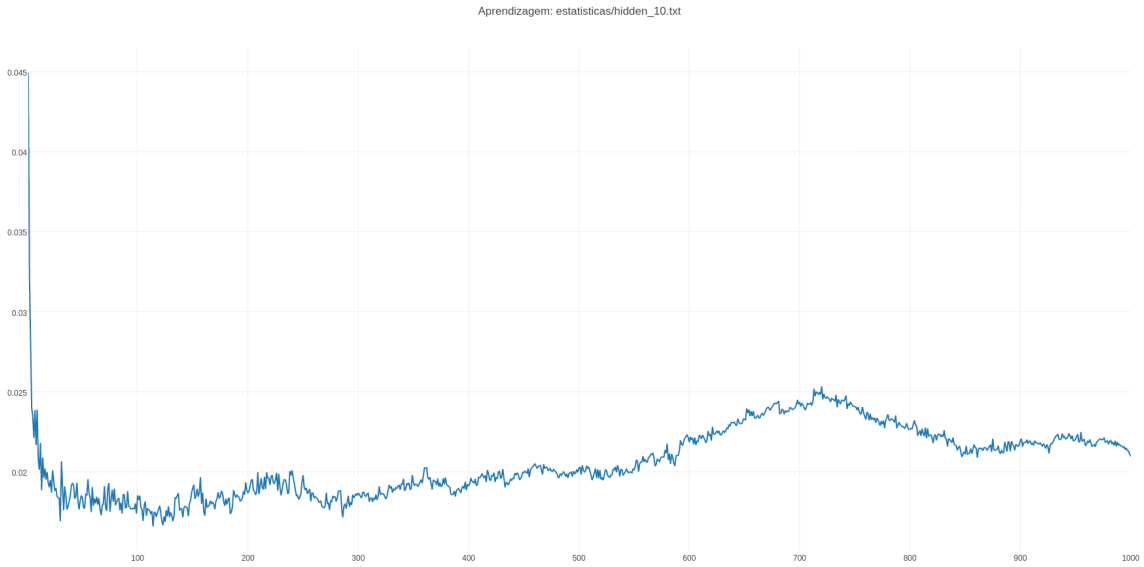

Figura 3.2: Variação do *Mean Squared Error* ao longo das épocas com 10 neurônios escondidos

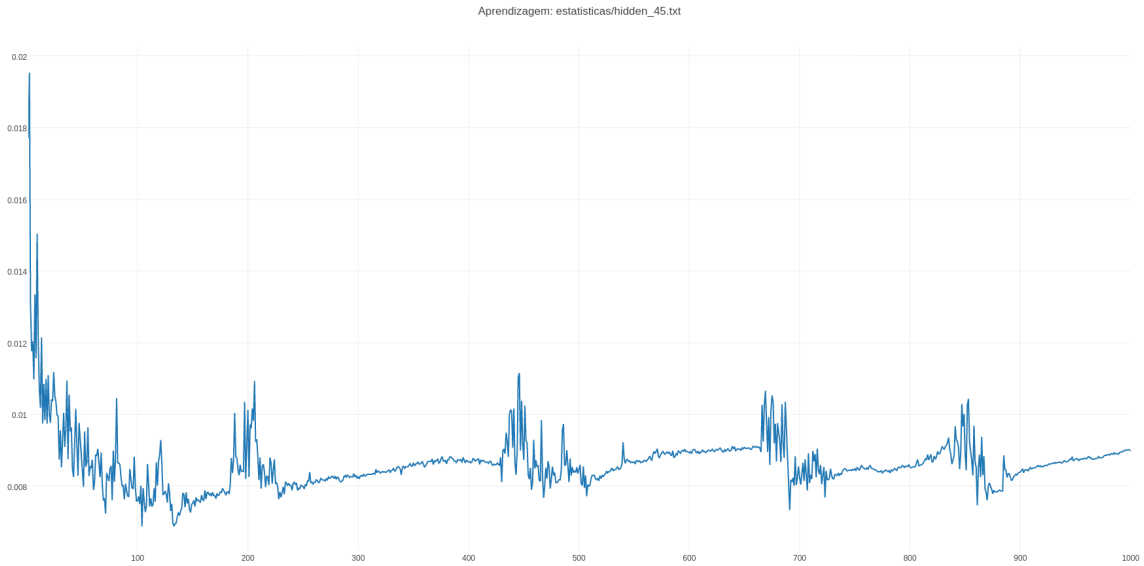

Figura 3.3: Variação do *Mean Squared Error* ao longo das épocas com 45 neurônios escondidos

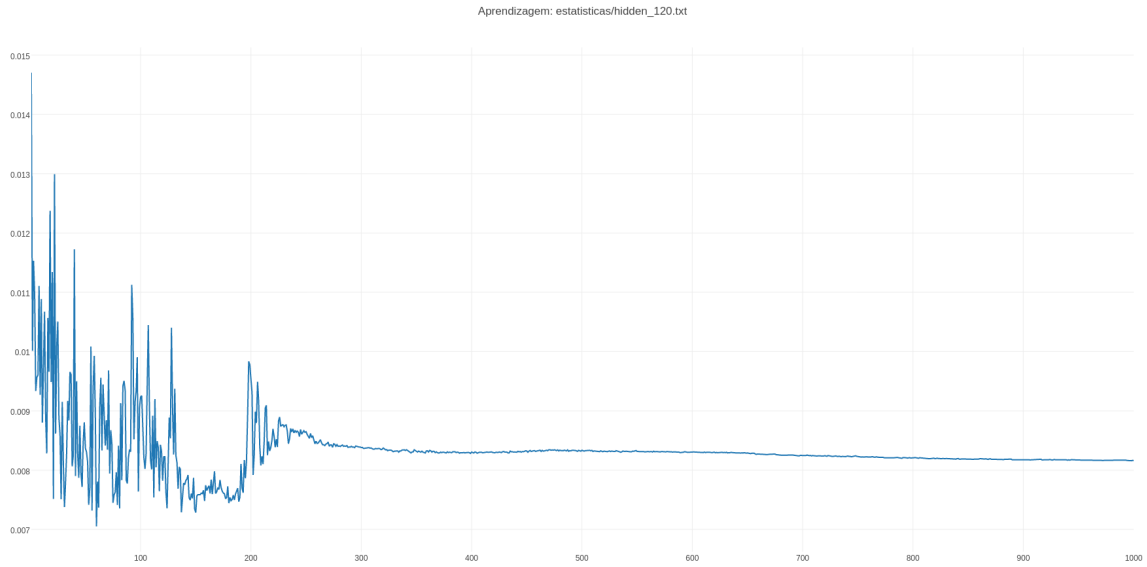

Figura 3.4: Variação do *Mean Squared Error* ao longo das épocas com 120 neurônios escondidos

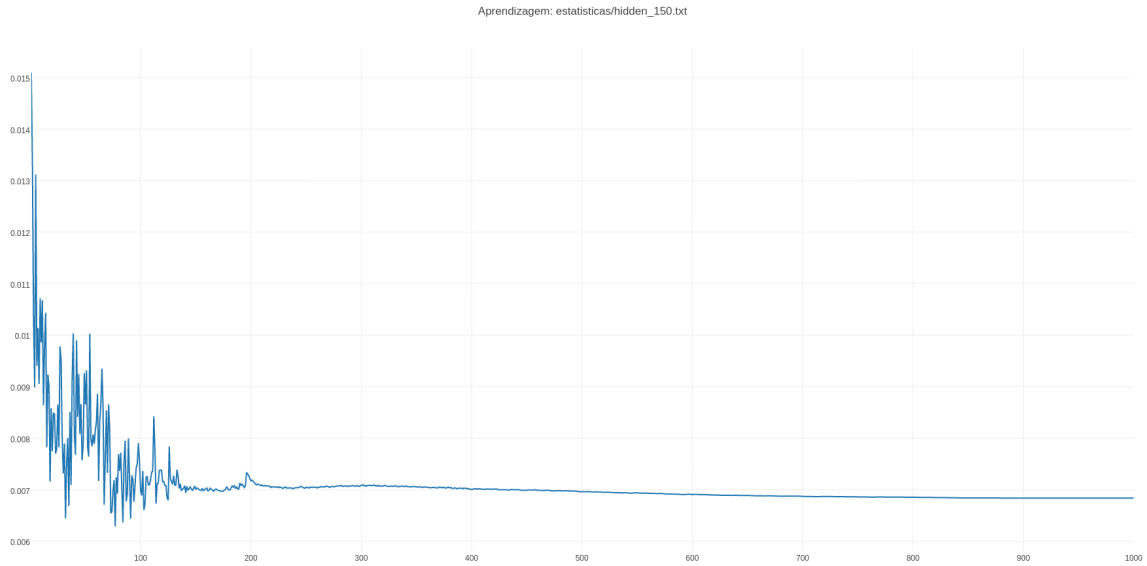

Figura 3.5: Variação do *Mean Squared Error* ao longo das épocas com 150 neurônios escondidos

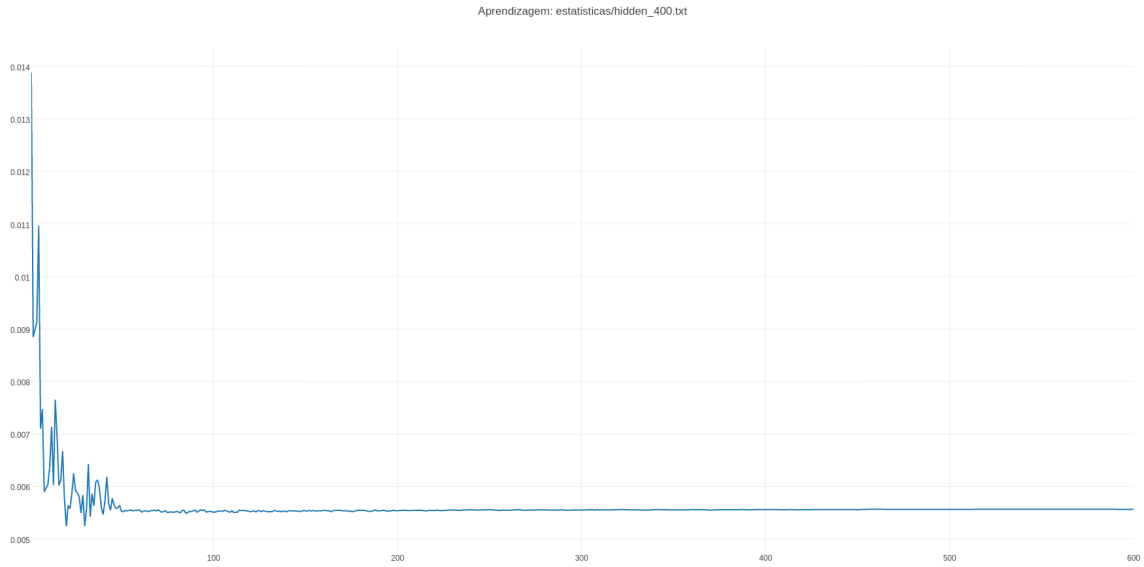

Figura 3.6: Variação do *Mean Squared Error* ao longo das épocas com 400 neurônios escondidos

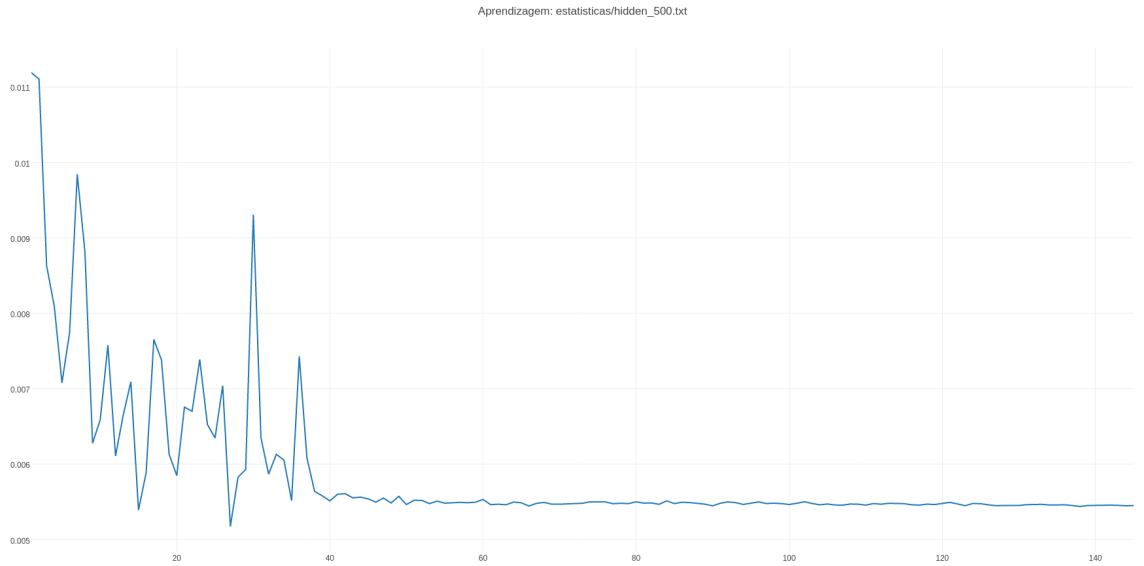

Figura 3.7: Variação do *Mean Squared Error* ao longo das épocas com 500 neurônios escondidos