

Como usar redes neurais

Gisele L Pappa
Computação Natural

Treinamento de MLPs

- Pode ser feito de 2 formas:
 - Online: logo após o exemplo ser apresentado a rede, os pesos são atualizados (mais comum)
 - Converge mais rápido
 - Batch: pesos são atualizados após uma época
 - Usa a média dos erros de cada instância para ajustar os pesos
 - Alcança erros menores
- Uma época equivale a apresentação de todos os exemplos de treinamento a rede

Treinamento

- A cada época, mude a ordem com que os exemplos são apresentados a rede
- Critério de parada
 - Convergência: Consideramos que o Back-prop converge quando a diferença absoluta entre os erros de uma época para outra é suficiente pequena (no intervalo $[0.1, 0.01]$).
 - Generalização – A cada época utilizar a rede em um segundo conjunto de dados

Generalização

- É influenciada por 3 fatores principais
 - O tamanho do conjunto de treinamento
 - A arquitetura da rede
 - A complexidade do problema
- Pode apresentar sérios problemas de overfitting (overtraining), memorizando os dados de treinamento (como no exemplo do exército americano)

Decisões/Parâmetros

- Como deve ser a entrada da rede?
- Como deve ser a saída da rede?
- Quantas camadas escondidas e neurônios?
- Quantos exemplos de treinamento?
- Que função de ativação?
- Como inicializar os pesos?
- Como escolher a taxa de aprendizagem?

Diferentes Tipos de Dados de Entrada

- Categórico
 - Azul, vermelho, verde
- Ordinal
 - Ratings de filmes: 1,2,3,4,5 estrelas
- Contínuo
 - Temperatura, pressão
- Periódico
 - Dias da semana

Variáveis Categóricas

- Codificação **local**
 - azul = 0.1, vermelho = 0.4, verde=0.8
- Codificação **distribuída**
 - azul = [1,0,0], vermelho = [0,1,0], verde=[0,1,1]
 - Não permite que vermelho esteja entre azul e verde
- Codificação **binária**
 - azul = [0,0], vermelho = [0,1], verde=[1,0]

Variáveis Ordinais

- *Ratings* de um filme
- Intervalos pré-definidos de idade (0-16,16-25,25-35,...)
- Variável tem uma ordem natural
- Utilizar uma codificação local, e tratar cada valor como um valor contínuo

Variáveis Periódicas

- Dias da semana
- Tem uma ordem, mas o maior e o menor são adjacentes
- Transforme em um ângulo
 - Seg = 0, Ter = $2\pi/7$, Quar = $4\pi/7$, Quin = $6\pi/7$,...
- Use o [cos,sen] do ângulo para codificar
 - Dom=[0.62,-0.78] Seg = [1,0] Ter = [0.62, 0.78]

Variáveis Contínuas

- Normalizar para que a média da amostra seja 0 e a variância 1

$$\mu_k = \frac{1}{N} \sum_{n=1}^N x_{nk} \quad \sigma_k^2 = \frac{1}{N-1} \sum_{n=1}^N (x_{nk} - \mu_k)^2$$

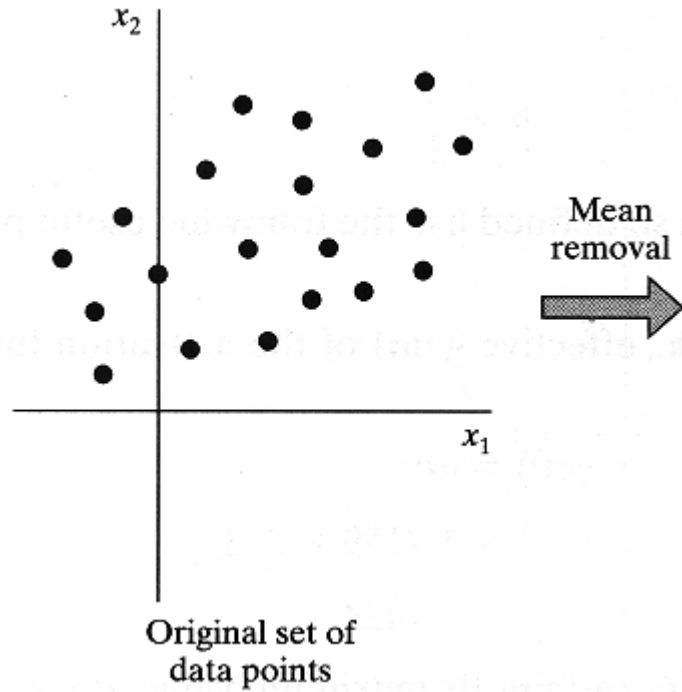
$$\hat{x}_{nk} = \frac{x_{nk} - \mu_k}{\sigma_k}$$

- Use apenas os dados de treinamento para calcular os parâmetros de normalização

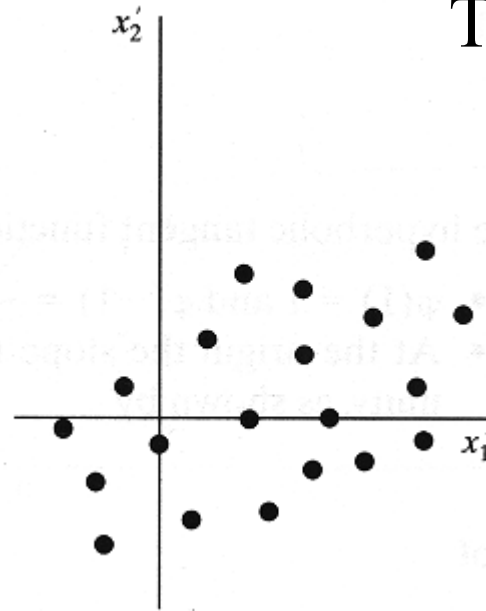

Como devem ser os dados de entrada?

- As variáveis de entrada devem ser não-correlacionadas
- As variáveis devem estar em uma escala apropriada

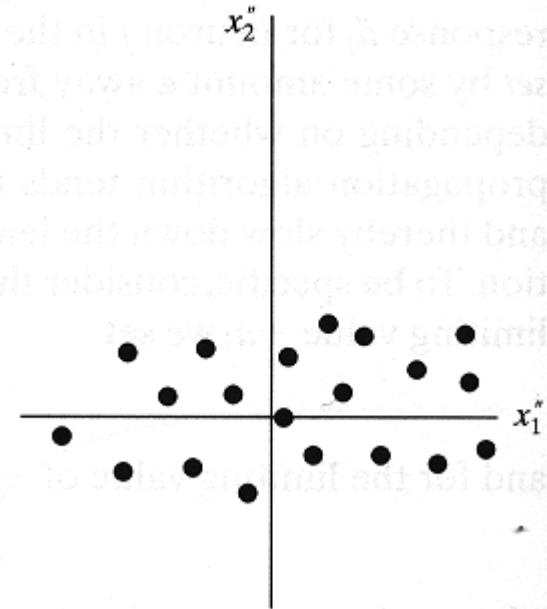
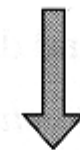
Transformações dos Dados de entrada



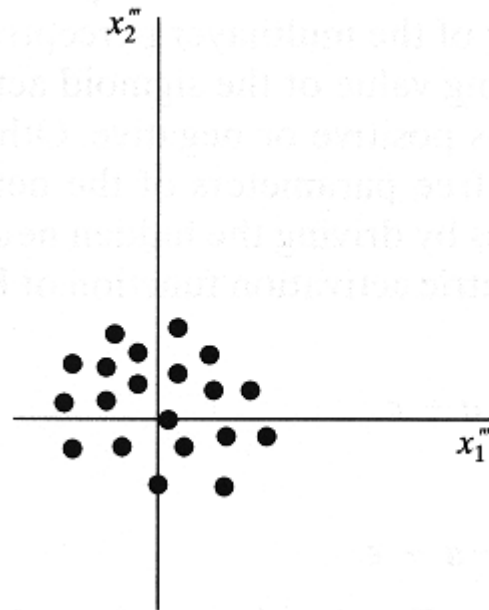
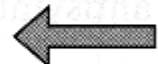
Mean
removal



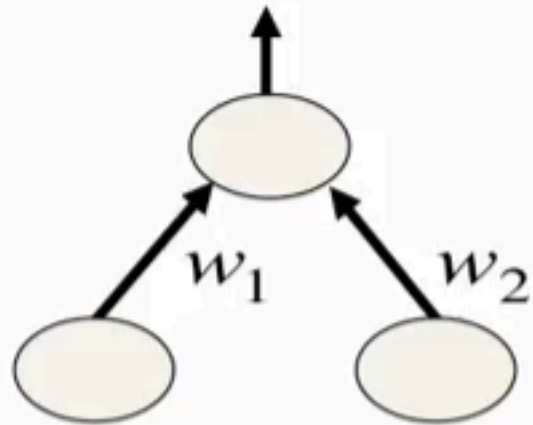
Decorrelation



Covariance
equalization



Scaling



color indicates
training case

101, 101 \rightarrow 2
101, 99 \rightarrow 0

gives error
surface



1, 1 \rightarrow 2
1, -1 \rightarrow 0

gives error
surface



Codificação da Saída

- Pode utilizar codificação distribuída, local, periódica ou binária
- Uma abordagem (ad hoc) é a saída variar entre $[0.1, 0.9]$ para evitar comportamentos estranhos nos extremos

Como usar as saídas para classificação?

- Codificação local: pegue a classes de codificação mais próxima
 - Ex: se as estrelas que o filme recebe são codificadas como $[0.1, 0.3, 0.5, 0.7, 0.9]$ e a saída da rede é 0.33, traduza para 2 estrelas
- Codificação distribuída: pegue a de maior valor
 - Ex: $[0.85, 0.15, 0.6]$, classe 1 deve ser escolhida
 - Pode também exigir que a classe escolhida seja maior que um limiar, ou a saída é desconhecida

Codificação da Saída

- Para ter certeza que a saída produz codificações razoáveis, pode-se transformar as saídas usando o *softmax*

$$\hat{a}_j = \frac{e^{a_j}}{\sum_{j'=1}^J e^{a_{j'}}}$$

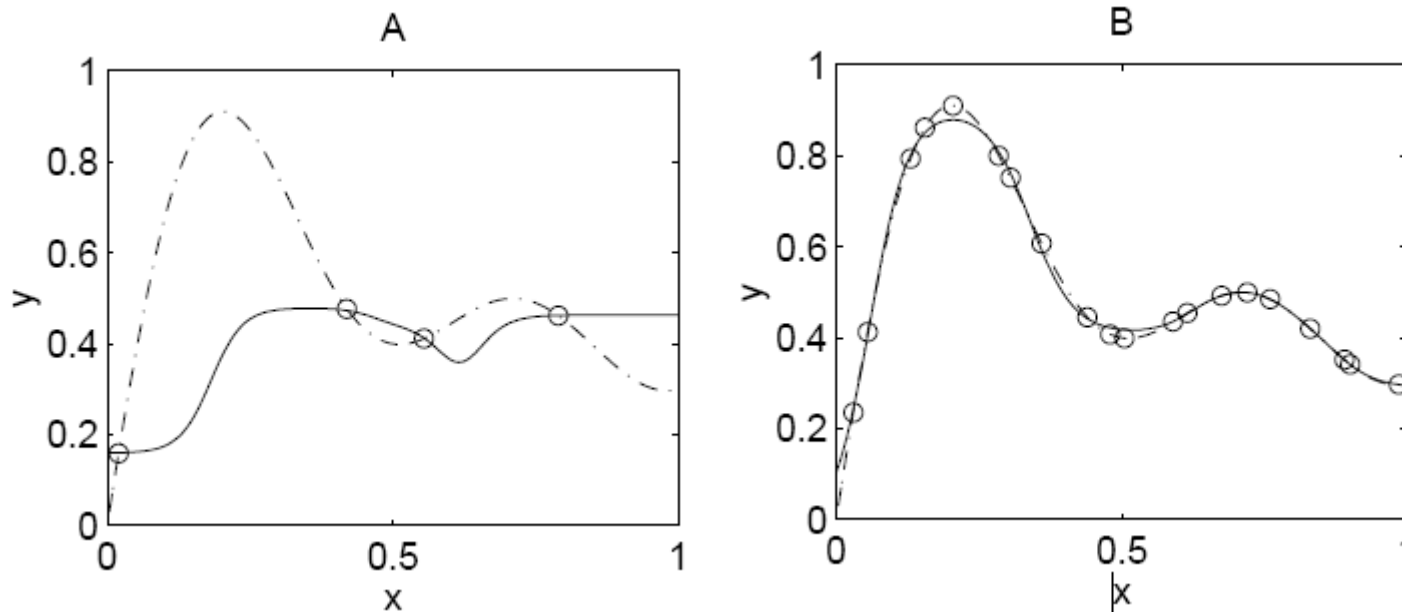
Valor de saída de um neurônio é normalizado pela soma das saídas de todos os neurônios

onde a_j corresponde a entrada normalizada pelo z-score

- Pode-se ler as saídas como probabilidades, que sempre somam 1.

Quantos exemplos de treinamento são necessários?

- Regra do dedão: o número de exemplos de treinamento deve ser de 5 a 10 vezes maior que o número de pesos sendo otimizados na rede

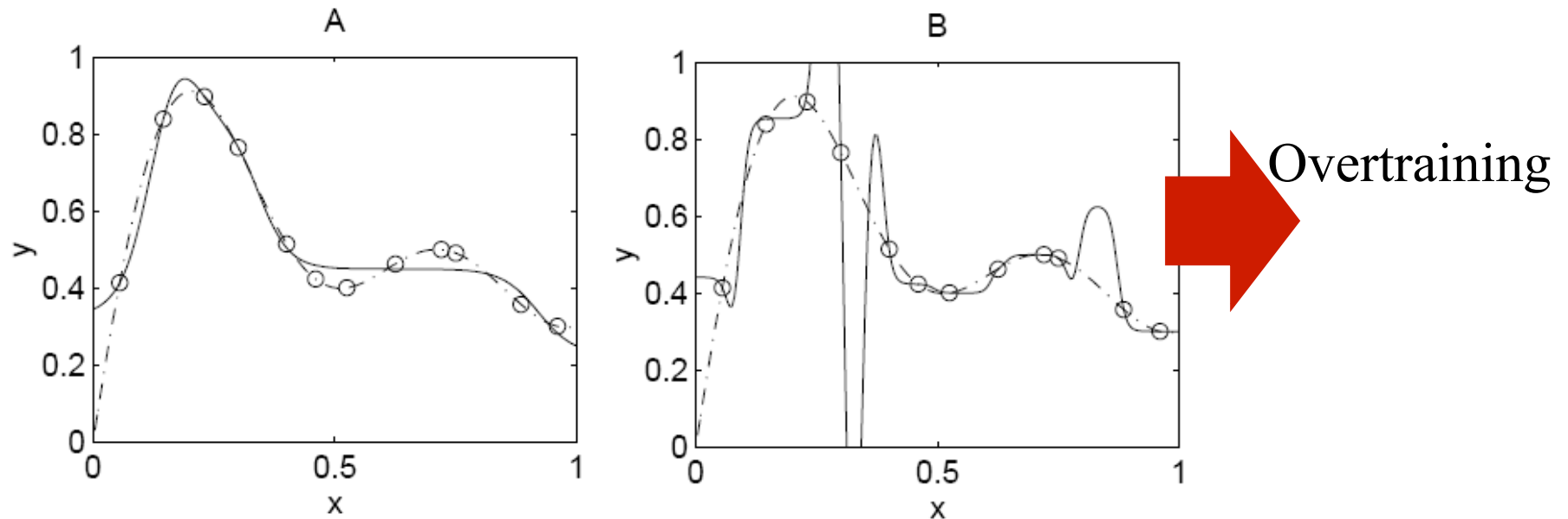


Aprendizagem com 4 versus 20 amostras

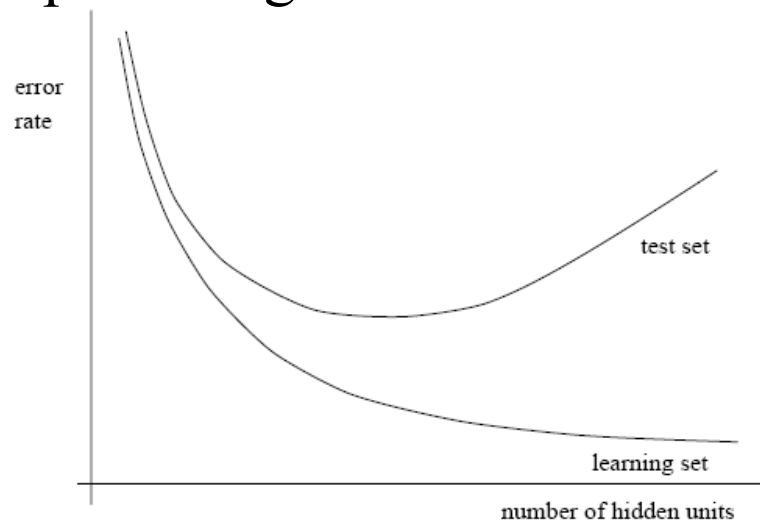
Quantas camadas e neurônios?

- Também depende da aplicação
- Uma ou duas camadas resolvem a maioria dos problemas
- Mas quantos neurônios devem ser usados em cada camada?
 - Aumentar o número de neurônios diminui o erro no treino (mas não necessariamente no teste)

Quantas camadas e neurônios?



Aprendizagem com 4 versus 20 neurônios na camada escondida

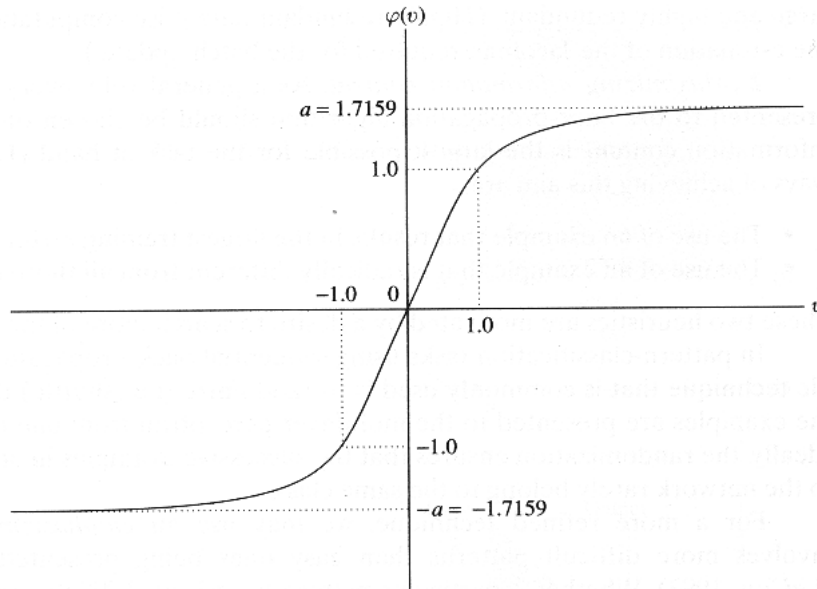


Erro no treino e no teste conforme aumentamos o número de neurônios

Que função de ativação utilizar?

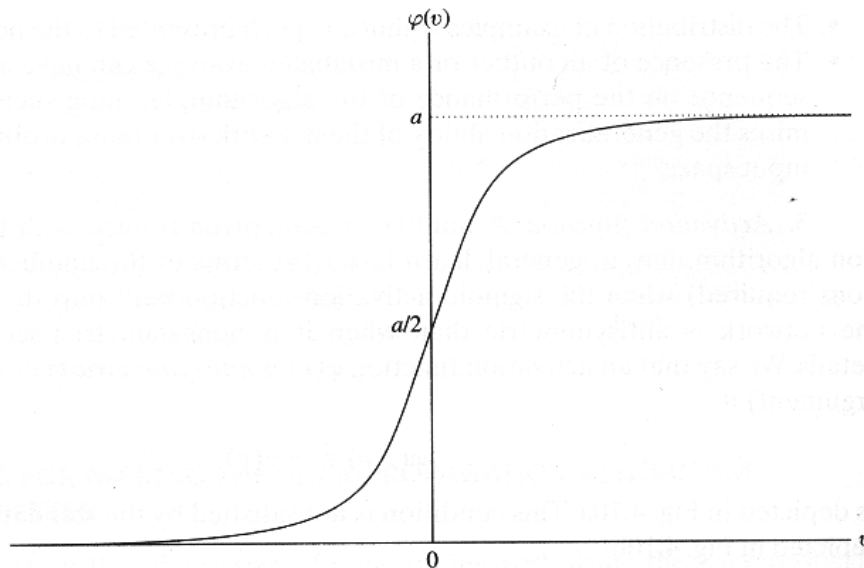
- Redes aprendem mais rápido com funções não simétricas
 - Sigmoides
 - Hiperbólica

Que função de ativação utilizar?



$$\varphi(v) = \frac{1}{1 + e^{-av}}$$

Sigmoide



$$\varphi(v) = a \tanh(bv)$$

Hiperbólica

Como inicializar os pesos?

- Se uma camada escondida tem um grande número de entradas, pequenas mudanças nos pesos podem causar uma mudança drástica na saída
 - Para evitar isso, normalmente usa-se pesos com valores pequenos quando o número de entradas é grande.
 - Inicializar os pesos de forma que sejam proporcionais a $\sqrt{\text{número de entradas}}$

Taxa de Aprendizagem

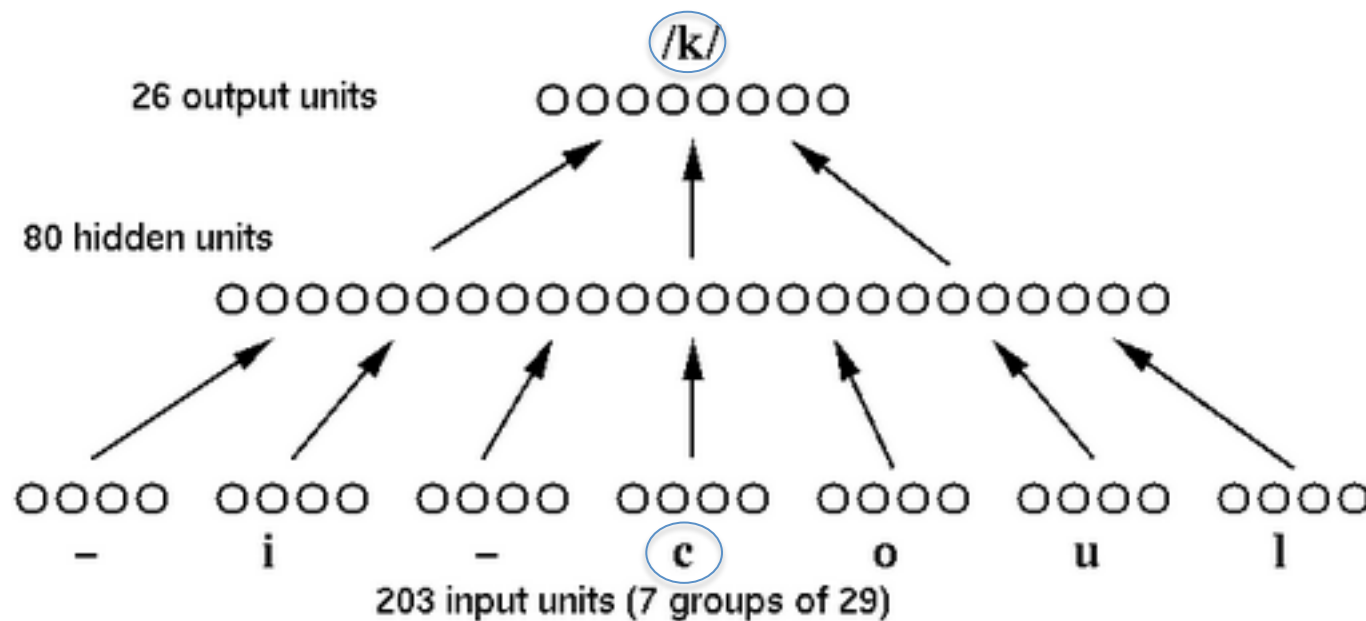
- Depende da aplicação, mas normalmente varia entre 0.1 e 0.9
- Algumas heurísticas variam esse valor durante o treinamento
- Pode ser determinada por $\sqrt{\text{número de entradas}}$

NetTalk

(Sejnowski e Rosenberg 1987)

- Objetivo:
 - A rede deve converter texto em fala
- Como
 - Dada uma palavra, fazer com que a rede aprenda os fonemas
- Rede utilizada
 - Backpropagation

Arquitetura



- Codificação distribuída
- O que está ao redor da letra dá contexto ao problema

Arquitetura

- 203 nós de entrada
 - 7 grupos (29 neurônios)
 - Cada grupo uma letra (26) + 3 para pontuação e separação da palavra
- 80 nós escondidos
- 26 nós de saída
 - Um fonema
 - 23 neurônios para características de articulação
 - 3 neurônios para identificar a tônica e separação de sílabas
- 18,629 conexões
- Função de ativação
 - sigmoide

Representação dos Fonemas

Table 1 Articulatory representation of phonemes and punctuations*

Symbol	Phoneme	Articulatory features
/a/	father	Low, Tensed, Central2
/b/	bet	Voiced, Labial, Stop
/c/	bought	Unvoiced, Velar, Medium
/d/	debt	Voiced, Alveolar, Stop
/e/	bake	Medium, Tensed, Front2
/f/	fin	Unvoiced, Labial, Fricative
/g/	guess	Voiced, Velar, Stop
/h/	head	Unvoiced, Glottal, Glide
/i/	Pete	High, Tensed, Front1
/k/	Ken	Unvoiced, Velar, Stop
/l/	let	Voiced, Dental, Liquid
/m/	met	Voiced, Labial, Nasal
/n/	net	Voiced, Alveolar, Nasal
/o/	boat	Medium, Tensed, Back2
/p/	pet	Unvoiced, Labial, Stop
/r/	red	Voiced, Palatal, Liquid
/s/	sit	Unvoiced, Alveolar, Fricative
/t/	test	Unvoiced, Alveolar, Stop
/u/	lute	High, Tensed, Back2
/v/	vest	Voiced, Labial, Fricative
/w/	wet	Voiced, Labial, Glide
/x/	about	Medium, Central2
/y/	yet	Voiced, Palatal, Glide

Dados

- Treinamento supervisionado
- 1024 palavras
- 54 fonemas
- 50 épocas

Erros

- Em certas ocasiões, a rede errava e substituía fonemas similares
 - Ex: thesis e these
- Nesse caso, a rede voltava a aprender o fonema correto muito mais rápido que da primeira vez

Resultados

- Depois de 10 épocas, palavras podiam ser entendidas
- Depois de 50 épocas
 - erro da rede no treinamento era de 5%
- Erro da rede no teste era de 22%
- Um outro sistema, baseado em tabelas, encontrava melhores resultados
- Porém, para criar as tabelas era necessário ter 10 linguistas trabalhando por um ano.
- Nettalk for produzido em 1 mês

Propriedades observadas

- Quanto mais palavras a rede aprende, melhor é seu poder de generalização quando ela é utilizada para pronunciar uma nova palavra
- Se algumas conexões da rede são perdidas, a performance geral da rede é pouco afetada (não existe um neurônio essencial, a aprendizagem é redistribuída)
- Re-aprender uma conexão perdida é muito mais rápido que aprender pela primeira vez

Leitura Recomendada

- Neural and Adaptive Systems: Fundamentals Through Simulations, Jose C. Principe , Neil R. Euliano, W. Curt Lefebvre, John Wiley & Sons, Inc.
- http://galaxy.agh.edu.pl/~vlsi/AI/backp_t_en/backprop.html