

## 第四章 感知器

在介绍人工神经网络之前，我们先来介绍下最简单的神经网络：感知器。

**感知器**，也是最简单的神经网络（只有一层）。感知器是由美国计算机科学家 Roseblatt 于 1957 年提出的。感知器可谓是最简单的人工神经网络，只有一个神经元。感知器也可以看出是线性分类器的一个经典学习算法。

感知器是对生物神经细胞的简单数学模拟。神经细胞也叫神经元（neuron），结构大致可分为细胞体和细胞突起。

- **细胞体**（Soma）中的神经细胞膜上有各种受体和离子通道，胞膜的受体可与相应的化学物质神经递质结合，引起离子通透性及膜内外电位差发生改变，产生相应的生理活动：兴奋或抑制。
- 细胞突起是由细胞体延伸出来的细长部分，又可分为树突和轴突。
  - **树突**（Dendrite）可以接受刺激并将兴奋传入细胞体。每个神经元可以有一或多个树突。
  - **轴突**（Axons）可以把兴奋从胞体传送到另一个神经元或其他组织。每个神经元只有一个轴突。

两个神经元之间或神经元与效应器细胞之间信息传递靠**突触**（Synapse）完成。突触是一个神经元的冲动传到另一个神经元或传到另一细胞间的相互接触的结构。

单个神经细胞可被视为一种只有两种状态的机器——兴奋和抑制。神经细胞的状态取决于从其它的神经细胞收到的输入信号量，及突触的强度（抑制或加强）。当信号量总和超过了某个阈值时，细胞体就会兴奋，产生电脉冲。电脉冲沿着轴突并通过突触传递到其它神经元。

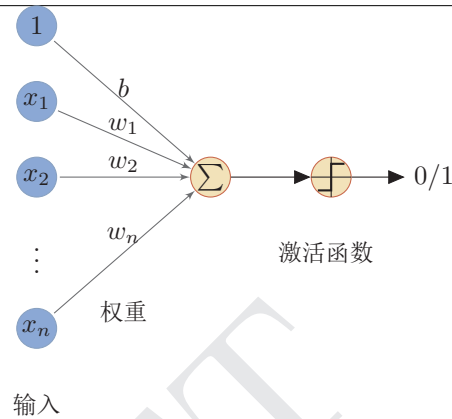


图 4.1: 感知器模型

感知器是模拟生物神经元行为的机器，有与生物神经元相对应的部件，如权重（突触）、偏置（阈值）及激活函数（细胞体），输出为0或1。

下面我们来介绍下感知器模型和学习算法。

## 4.1 两类感知器

图4.1给出了感知器模型的结构。

给定一个  $n$  维的输入  $\mathbf{x} = (x_1, x_2, \dots, x_n)$ ,

$$\hat{y} = \begin{cases} +1 & \text{当 } \mathbf{w}^T \mathbf{x} + b > 0 \\ -1 & \text{当 } \mathbf{w}^T \mathbf{x} + b \leq 0 \end{cases}, \quad (4.1)$$

其中， $\mathbf{w}$  是  $n$  维的权重向量， $b$  是偏置。 $\mathbf{w}$  和  $b$  是未知的，需要从给定的训练数据集中学习得到。

不失一般性，我们使用增广的输入和权重向量，公式4.1可以简写为：

$$\hat{y} = \begin{cases} +1 & \text{当 } \mathbf{w}^T \mathbf{x} > 0 \\ -1 & \text{当 } \mathbf{w}^T \mathbf{x} \leq 0 \end{cases}, \quad (4.2)$$

接下来我们看下感知器是如何学习的。

### 4.1.1 感知器学习算法

给定  $N$  个样本的训练集:  $(\mathbf{x}_i, y_i), i = 1, \dots, N$ 。我们希望学习到参数  $\mathbf{w}^*$ , 使得

$$\begin{aligned} \mathbf{w}^{*T} \mathbf{x}_i &> 0 \quad \text{当 } y_i > 0, \\ \mathbf{w}^{*T} \mathbf{x}_i &< 0 \quad \text{当 } y_i < 0. \end{aligned} \quad (4.3)$$

公式4.3等价于  $\mathbf{w}^{*T}(y_i \mathbf{x}_i) > 0$ 。

Rosenblatt [1958] 首次提出了感知器的学习算法。这个算法是错误驱动的在线学习算法。先初始化一个权重向量  $\mathbf{w}_0$  (通常是全零向量), 然后每次分错一个样本时, 就用这个样本来更新权重。具体的学习过程如算法5.2所示。

#### 算法 4.1: 两类感知器算法

```

输入: 训练集:  $(\mathbf{x}_i, y_i), i = 1, \dots, N$ , 迭代次数:  $T$ 
输出:  $\mathbf{w}$ 

1 初始化:  $\mathbf{w}_0 = 0$ ;
2  $k = 0$ ;
3 for  $t = 1 \dots T$  do
4   for  $i = 1 \dots N$  do
5     选取一个样本  $(\mathbf{x}_i, y_i)$ , if  $\mathbf{w}^T(y_i \mathbf{x}_i) < 0$  then
6        $\mathbf{w}_{k+1} = \mathbf{w}_k + y_i \mathbf{x}_i$ ;
7        $k = k + 1$ ;
8     end
9   end
10 end
11 return  $\mathbf{w}_k$ ;
    
```

### 4.1.2 收敛性证明

Novikoff [1963] 证明对于两类问题, 如果训练集是线性可分的, 那么感知器算法可以在有限次迭代后收敛。然而, 如果训练集不是线性分隔的, 那么这个算法则不能确保会收敛。

**定义 4.1 – 两类线性可分：** 对于训练集  $\mathcal{D} = \{(x_i, y_i) \mid y_i \in \{-1, 1\}\}_{i=1}^n$ ，如果存在一个正的常数  $\gamma (\gamma > 0)$  和权重向量  $\mathbf{w}^*$ ，并且  $\|\mathbf{w}^*\| = 1$ ，对所有  $i$  都满足  $(\mathbf{w}^*)^T (y_i \mathbf{x}_i) > \gamma$  ( $\mathbf{x}_i \in \mathbb{R}^m$  为样本  $x_i$  的增广特征向量)，那么训练集  $\mathcal{D}$  是线性可分的。

在数据集是两类线性可分的条件下，我们可以证明如下定理。

**定理 4.1 – 感知器收敛性：** 对于任何线性可分的训练集  $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ ，假设  $R$  是所有样本中输入向量的模的最大值。

$$R = \max_i \|x_i\|$$

那么在感知器学习算法5.2中，总共的预测错误次数  $K < \frac{R^2}{\gamma^2}$ 。

证明如下：

权重向量的更新公式为：

$$\mathbf{w}_k = \mathbf{w}_{k-1} + y_k \mathbf{x}_k, \quad (4.4)$$

这里， $\mathbf{x}_k, y_k$  为第  $k$  个错误分类的样本。

因为初始权重向量为0，因此在第  $K$  次更新时，

$$\mathbf{w}_K = \sum_{k=1}^K y_k \mathbf{x}_k. \quad (4.5)$$

那么，

(1)  $\|\mathbf{w}_K\|^2$  的上界为：


$$\begin{aligned} \|\mathbf{w}_K\|^2 &= \left\| \sum_{k=1}^K y_k \mathbf{x}_k \right\|^2 \\ &\leq \sum_{k=1}^K \|y_k \mathbf{x}_k\|^2 \end{aligned}$$

$$\begin{aligned} &\leq \sum_{k=1}^K \max_{k=1}^K \|y_k \mathbf{x}_k\|^2 \\ &\leq K \cdot R^2 \end{aligned} \quad (4.6)$$

(2) 我们再来看下  $\|\mathbf{w}_K\|^2$  的下界。

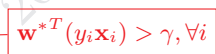
首先，因为两个向量内积的平方一定小于等于这两个向量的模的乘积。

$$\|\mathbf{w}^{*T} \mathbf{w}_K\|^2 \leq \|\mathbf{w}^*\|^2 \cdot \|\mathbf{w}_K\|^2 = \|\mathbf{w}_K\|^2. \quad (4.7)$$



因此，

$$\begin{aligned} \|\mathbf{w}_K\|^2 &\geq \|\mathbf{w}^{*T} \mathbf{w}_K\|^2 \\ &= \|\mathbf{w}^{*T} \sum_{k=1}^K (y_k \mathbf{x}_k)\|^2 \\ &= \left\| \sum_{k=1}^K \mathbf{w}^{*T} (y_k \mathbf{x}_k) \right\|^2 \\ &\geq K^2 \gamma^2 \end{aligned} \quad (4.8)$$



由公式4.6和4.8，得到

$$K^2 \gamma^2 \leq \|\mathbf{w}_K\|^2 \leq K \times R^2 \quad (4.9)$$

取最左和最右的两项，进一步得到， $K^2 \gamma^2 \leq K \cdot R^2$ 。然后两边都除  $K$ ，最终得到

$$K \leq \frac{R^2}{\gamma^2}. \quad (4.10)$$

因此，在线性可分的条件下，算法5.2会在  $\frac{R^2}{\gamma^2}$  步内收敛。如果训练集不是线性可分的，就永远不会收敛 [Freund and Schapire, 1999]。

## 4.2 多类感知器

原始的感知器的输出是0或1，不能提供概率形式的输出，因此只能处理两类问题。为了使得感知器能处理多类问题以及更复杂的结构化学习任务，我们引入一个特征函数  $\phi(x, y)$  将输入输出对映射到一个向量空间中 [Collins, 2002]。这样，我们可以得到一个更为泛化的感知器：

$$\hat{y} = \arg \max_{y \in \text{Gen}(\mathbf{x})} \mathbf{w}^T \phi(x, y), \quad (4.11)$$

这里  $\text{Gen}(x)$  表示输入  $x$  所有的输出目标集合。当处理  $C$  类分类问题时， $\text{Gen}(x) = \{1, \dots, C\}$ 。

在上面几节中，分类函数都是在输入  $x$  的向量空间上。通过引入特征函数  $\phi(x, y)$ ，感知器不但可以用于多类分类问题，也可以用于结构化学习问题，比如输出是序列或来其它结构化的形式。

当  $y$  为离散变量时 ( $y \in \{1, \dots, C\}$ )，类别也可以表示为向量：

$$\phi(y = c) = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \boxed{1} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \leftarrow \text{第 } c \text{ 行}$$

如果每个样本可以有多个类别，标签分类  $y = \{c, k\}$  时，类别可以表示为向量：

$$\phi(y = \{c, k\}) = \begin{bmatrix} 0 \\ \vdots \\ \boxed{1} \\ \vdots \\ \boxed{1} \\ \vdots \\ 0 \end{bmatrix} \quad \begin{array}{l} \leftarrow \text{第 } c \text{ 行} \\ \leftarrow \text{第 } k \text{ 行} \end{array}$$

$\phi(\mathbf{x}, y)$  可以看成是  $\phi(y)$  和  $\mathbf{x}^T$  的乘积得到矩阵的向量化。

$$\phi(\mathbf{x}, y = c) = \text{vec}(\phi(y)\mathbf{x}^T), \quad (4.12)$$

这里， $\text{vec}$  是向量化算子。

$$\phi(x, y = c) = \begin{bmatrix} \vdots \\ 0 \\ \boxed{\mathbf{x}} \\ 0 \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ 0 \\ \boxed{x_1} \\ \vdots \\ \boxed{x_m} \\ 0 \\ \vdots \end{bmatrix} \quad \begin{array}{l} \leftarrow \text{第 } c * m + 1 \text{ 行} \\ \leftarrow \text{第 } c * m + m \text{ 行} \end{array} \quad (4.13)$$

多类感知器算法的训练过程如算法4.2所示。

#### 算法 4.2: 多类感知器算法

输入: 训练集:  $(x_i, y_i), i = 1, \dots, N$ , 最大迭代次数:  $T$

输出:  $\mathbf{w}_k$

```

1  $\mathbf{w}_0 = 0$  ;
2  $k = 0$  ;
3 for  $t = 1 \dots T$  do
4   for  $i = 1 \dots N$  do
5     选取一个样本  $(x_i, y_i)$ ;
6     用公式4.11计算预测类别  $\hat{y}_i$ ;
7     if  $\hat{y}_i \neq y_i$  then
8        $\mathbf{w}_{k+1} = \mathbf{w}_k + (\phi(\mathbf{x}_i, y_i) - \phi(\mathbf{x}_i, \hat{y}_i))$ ;
9        $k = k + 1$  ;
10    end
11  end
12 end
13 return  $\mathbf{w}_k$  ;
    
```

#### 4.2.1 多类感知器的收敛性

多类分类时，感知器的收敛情况。Collins [2002] 给出了多类感知器在多类线性可分的收敛性证明，具体推导过程和两类分类器比较类似。

**定义 4.2 – 多类线性可分:** 对于训练集  $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ ，如果存在一个正的常数  $\gamma (\gamma > 0)$  和权重向量  $\mathbf{w}^*$ ，并且  $\|\mathbf{w}^*\| = 1$ ，对所有  $i$  都满足  $\langle \mathbf{w}^*, \phi(x_i, y_i) \rangle - \langle \mathbf{w}^*, \phi(x_i, y) \rangle > \gamma, y \neq y_i$  ( $\phi(x_i, y_i) \in \mathbb{R}^m$  为样本  $x_i, y_i$  的特征向量)，那么训练集  $\mathcal{D}$  是线性可分的。

**定理 4.2 – 多类感知器收敛性:** 对于任何线性可分的训练集  $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ ，假设  $R$  是所有样本中错误类别和真实类别在特征空



间  $\phi(x, y)$  最远的距离。

$$R = \max_i \max_{z \neq y_i} \|\phi(x_i, y_i) - \phi(x_i, z)\|.$$

那么在多类感知器学习算法总共的预测错误次数  $K < \frac{R^2}{\gamma^2}$ 。

### 4.3 投票感知器

从定理4.2可以看出，如果训练数据是线性可分的，那么感知器可以找到一个判别函数来分割不同类的的数据。并且如果边际距离越大，收敛越快。但是，感知器并不能保证找到的判别函数是最优的（比如泛化能力高），这样可能导致过拟合。

此外，从感知器的迭代算法可以看出：在迭代次序上排在后面的错误点，比前面的错误点对最终的权重向量影响更大。比如有 1,000 个训练样本，在迭代 100 个样本后，感知器已经学习到一个很好的权重向量。在接下来的 899 个样本上都预测正确，也没有更新权重向量。但是在最后第 1,000 个样本时预测错误，并更新了权重。这次更新可能反而使得权重向量变差了。

为了这种情况，可以使用“参数平均”的策略来提高感知器的鲁棒性，也叫投票感知器（Voted Perceptron）[Freund and Schapire, 1999]。

投票感知器记录第  $k$  次更新后得到的权重  $\mathbf{w}_k$  在其后分类中正确分类样本的次数  $c_k$ 。这样最后的分类器形式为（假设输出为 0 或 1）：

$$\hat{y} = \text{sign}\left(\sum_{k=1}^K c_k \text{sign}(\mathbf{w}_k^T x)\right). \quad (4.14)$$

投票感知器虽然提高了感知器的泛化能力，但是需要保存  $K$  个权重向量。在实际操作中会带来额外的开销。因此，人们经常会使用一个简化的版本，也叫做平均感知器（Averaged Perceptron）[Collins, 2002]。

$$\begin{aligned} \hat{y} &= \text{sign}\left(\sum_{k=1}^K c_k (\mathbf{w}_k^T x)\right) \\ &= \text{sign}\left(\sum_{k=1}^K c_k \mathbf{w}_k\right)^T x \\ &\equiv \text{sign}(\bar{\mathbf{w}}^T x), \end{aligned} \quad (4.15)$$

其中,  $\bar{\mathbf{w}}$  为平均的权重向量。

假设  $\mathbf{w}^{t,i}$  是在第  $t$  轮更新到第  $i$  个样本时权重向量的值, 平均的权重向量  $\bar{\mathbf{w}}$  也可以写为

$$\bar{\mathbf{w}} = \frac{\sum_{t=1}^T \sum_{i=1}^n \mathbf{w}^{t,i}}{nT} \quad (4.16)$$

这个方法非常简单, 只需要在算法4.2中增加一个  $\bar{\mathbf{w}}$ , 并且在处理每一个样本后, 更新

$$\bar{\mathbf{w}} = \bar{\mathbf{w}} + \mathbf{w}^{t,i} \quad (4.17)$$

这里要注意的是,  $\bar{\mathbf{w}}$  需要在处理每一个样本时都需要更新, 并且  $\bar{\mathbf{w}}$  和  $\mathbf{w}^{t,i}$  都是稠密向量。因此, 这个操作比较费时。为了提高迭代速度, 有很多改进的方法, 让这个更新只需要在错误预测发生时才进行更新。

算法4.3给出了一个改进的平均感知器算法的训练过程 [Daumé III]。

#### 算法 4.3: 平均感知器算法

输入: 训练集:  $(x_i, y_i), i = 1, \dots, N$ , 最大迭代次数:  $T$

输出:  $\bar{\mathbf{w}}$

```

1  $\mathbf{w} = 0$  ;
2  $\mathbf{u} = 0$  ;
3  $c = 0$  ;
4 for  $t = 1 \dots T$  do
5   for  $i = 1 \dots N$  do
6     选取一个样本  $(x_i, y_i)$ ;
7     用公式4.11计算预测类别  $\hat{y}_t$ ;
8     if  $\hat{y}_t \neq y_t$  then
9        $\mathbf{w} = \mathbf{w} + ((x_t, y_t) - (x_t, \hat{y}_t))$ ;
10       $\mathbf{u} = \mathbf{u} + c \cdot ((x_t, y_t) - (x_t, \hat{y}_t))$ ;
11    end
12     $c = c + 1$  ;
13  end
14 end
15  $\bar{\mathbf{w}} = \mathbf{w}_T - \frac{1}{c} \mathbf{u}$  ;
16 return  $\bar{\mathbf{w}}$  ;
```

## 4.4 总结和深入阅读

Rosenblatt [1958] 最早提出了两类感知器算法，并随后给出了感知机收敛定理。但是感知器的输出是离散的以及学习算法比较简单，不能解决线性不可分问题，限制了其应用范围。Minsky and Seymour [1969] 分析了感知机的局限性，证明感知机不能解决非常简单的异或（XOR）问题。虽然他也认为多层的网络可以解决非线性问题，但是遗憾的是，在当时这个问题还不可解。直到1980年以后，Geoffrey Hinton、Yann LeCun 等人用连续输出代替离散的输出，并将反向传播算法（Backpropagation, BP）[Werbos, 1974] 引入到多层感知器 [Williams and Hinton, 1986]，人工神经网络才又重新引起人们的注意。Minsky and Papert [1987] 也修正之前的看法。

另外一方面，人们对感知器本身的认识也在不断发展。Freund and Schapire [1999] 提出了使用核技巧改进感知器学习算法，并用投票感知器来提高泛化能力。Collins [2002] 将感知器算法扩展到结构化学习，给出了相应的收敛性证明，并且提出一种更加有效并且实用的参数平均化策略。[McDonald et al., 2010] 又扩展了平均感知器算法，使得感知器可以在分布式计算环境中并行计算，这样感知器可以用在大规模机器学习问题上。

## 参考文献

- James A Anderson and Edward Rosenfeld. *Talking nets: An oral history of neural networks*. MiT Press, 2000.
- Yoshua Bengio. Learning deep architectures for AI. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.
- Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*, 5(2):157–166, 1994.
- Yoshua Bengio, Jean-Sébastien Senécal, et al. Quick training of probabilistic neural nets by importance sampling. In *AISTATS Conference*, 2003.
- Yoshua Bengio, Nicolas Boulanger-Lewandowski, and Razvan Pascanu. Advances in optimizing recurrent networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 8624–8628. IEEE, 2013.
- C.M. Bishop. *Pattern recognition and machine learning*. Springer New York., 2006.
- P.F. Brown, P.V. Desouza, R.L. Mercer, V.J.D. Pietra, and J.C. Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479, 1992.
- Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.
- Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

- Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, 2002.
- Hal Daumé III. A course in machine learning. <http://ciml.info/>. [Online].
- John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159, 2011.
- R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley, New York, 2nd edition, 2001. ISBN 0471056693.
- Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating second-order functional knowledge for better option pricing. *Advances in Neural Information Processing Systems*, pages 472–478, 2001.
- Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.
- Yoav Freund and Robert E Schapire. Large margin classification using the perceptron algorithm. *Machine learning*, 37(3):277–296, 1999.
- Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International conference on artificial intelligence and statistics*, pages 249–256, 2010.
- Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 315–323, 2011.
- Ian Goodfellow, Aaron Courville, and Yoshua Bengio. Deep learning. Book in preparation for MIT Press, 2015. URL <http://goodfeli.github.io/dlbook/>.
- T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2001.
- Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, 29(6):82–97, 2012.
- Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

- Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, and Jürgen Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.
- David H Hubel and Torsten N Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology*, 195(1):215–243, 1968.
- Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.
- M.I. Jordan. *Learning in Graphical Models*. Kluwer Academic Publishers, 1998.
- Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- Ryan McDonald, Keith Hall, and Gideon Mann. Distributed training strategies for the structured perceptron. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 456–464. Association for Computational Linguistics, 2010.
- Marvin Minsky and Papert Seymour. Perceptrons. 1969.
- Marvin L Minsky and Seymour A Papert. *Perceptrons - Expanded Edition: An Introduction to Computational Geometry*. MIT press Boston, MA:, 1987.
- T.M. Mitchell. *Machine learning*. Burr Ridge, IL: McGraw Hill, 1997.
- Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic language models. *arXiv preprint arXiv:1206.6426*, 2012.
- Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.
- Albert BJ Novikoff. On convergence proofs for perceptrons. Technical report, DTIC Document, 1963.
- F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386–408, 1958.

- David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5:3, 1988.
- Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.
- Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014.
- Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. Phoneme recognition using time-delay neural networks. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 37(3):328–339, 1989.
- Paul Werbos. Beyond regression: New tools for prediction and analysis in the behavioral sciences. 1974.
- Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.
- DE Rumelhart GE Hinton RJ Williams and GE Hinton. Learning representations by back-propagating errors. *Nature*, pages 323–533, 1986.
- Matthew D Zeiler. Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.