保密类别_____ 编　号 744058

毕　业　论　文

基于神经网络的语音增强方法研究

学部(院)　信息工程学院

专　　业　电子信息工程

班　　级　一班

姓　　名　G4Y8u9

指导教师　Pebble

中 国 传 媒 大 学

2018 年 5 月 17 日

基于神经网络的语音增强方法研究

G4Y8u9

摘　要

近年来，随着计算机技术的普及与进步，语音消息已经成为人们日常生活中非常重要、常见的消息类型。同时，人们对于语音消息质量的需求也在逐步提升，这使得信号处理领域出现了许多种语音增强方法。这些方法大都是先在带噪语音的频域上估计噪声的频谱信息，然后将噪声谱减掉，从而得到纯净语谱的预测值。但由于噪声的一些客观属性的存在，比如随机性与突变型，使这些传统的语音增强方法缺点也很明显——难以准确估计噪声，甚至对原有语音信息造成破坏。另外，传统的语音增强方法对于非平稳噪声处理并不理想，而在现实生活中，非平稳噪声又极为常见。

而最近机器学习（Machine Learning）理论、技术方兴未艾，基于人工神经网络（Neural Network）的语音增强方法被一些研究人员提出。神经网络的“黑箱”特性在有大量训练集数据支持的条件下，理论上可以充分学习带噪语音与纯净语音之间的某种复杂映射关系。同时，之前提到的非平稳噪声也可以被神经网络“记住”，使其相比于传统语音增强方法，神经网络对非平稳噪声的处理更加游刃有余。本论文中，我们将搜集语音数据与多种噪声数据，生成带噪语音、纯净语音的训练集，之后搭建各种类型的神经网络并训练之，研究神经网络的多种属性对输出结果的影响。

首先我们将使用基于多种噪声类型的语音作为训练集，将带噪语音作短时傅里叶变换（Short-Time Fourier Transform, STFT）得到语音数据的语谱图，将数据取对数、高斯归一化，再将每帧数据与其两边一些帧数据为一帧组作为神经网络的输入，然后搭建深层神经网络（Deep Neural Network, DNN）开始训练。

另外，我们的目标是使输出结果令人类满意，所以我们取语音信号的梅尔频率（Mel Frequency）数据加入到网络的输入端，梅尔刻度（Mel Scale）是基于人耳对音高主观感受而定的非线性频率刻度，因此更能概括一条语音信号的特征[1][2]，然后我们将定量分析改进如何。

我们还会介绍其他网络类型，比如递归神经网络（Recurrent Neural Network, RNN），想到这种神经网络的原因，一方面考虑到我们希望神经网络“记住”一些非平稳噪声的模式，很多非平稳噪声的在语谱图上会呈现出很多规则的图案，而 CNN 非常善于识别图案的学习任务；另一方面，语音信号前后有非常强的关联性，虽然之前 DNN 的方法

中已经对语谱数据编以帧组，但其对时间的关联性有限，而 RNN 可以接受更广泛的时间序列结构输入，尤其某些特殊类型的 RNN，比如长短期记忆网络（Long Short-Term Memory, LSTM）与门控递归单元（Gated Recurrent Unit, GRU），也被一些语音处理的研究人员提出并应用。

关键词：语音增强，神经网络

SPEECH ENHANCEMENT BASED ON NEURAL NETWORK

G4Y8u9

ABSTRACT

In the recent years, with the improving and popularization of the computer technology, speech messages have been one of the most important and common types of messages. At the same time, the requirement of the speech messages' quality of people is also increasing, which results in lots of speech enhancement methods thriving in the field of signal processing. Most of those methods begin with estimating the frequency information of noise based on the noisy speech signal's frequency domain, and subtract the spectrum of noise, then we will have the prediction of pure speech signal's spectrum. However, due to the existence of some objective properties of noise, such as randomness and mutability, those classical speech enhancement methods' disadvantage is also obvious, i.e., it's difficult to estimate noisy accurately, and causing even damage to original speech information. Additionally, classic speech enhancement methods' processing result with regard to non-stationary noise is not ideal, though non-stationary noise is pretty common in real life.

The theory and technology of machine learning are just unfolding, leading to the suggest of speech enhancement based on artificial neural network. Due to neural network's "black box" property, it can fully learn some complicate map relation between pure speech signal and noisy one with the support of sufficient train data theoretically. In the meantime, the non-stationary noise mentioned earlier can also be "remembered" by neural network, which makes neural network is better at processing non-stationary noise compared with classical speech enhancement methods. In this paper, we're going to collect multi-type of speech data and noise data, and generate noisy speech and pure speech as training data, then construct types of neural networks and train them, and research in the influences of different properties of neural networks.

First, it is only white noise that will be used as training data, and obtain its spectrum by short-time Fourier transform (STFT), then calculate its log value and Gaussian normalize it, and combine every frame with its neighbors as the input of neural work. Then we construct deep neural network and start training.

Then we will upgrade the process. First we will use multi-type of noisy data as training data. Second, since out target is to make the output satisfy humans, we decide to calculate Mel frequency and add to the input of the network. Mel scale is based on the human ears subjective pitch feeling's non-linear scale, hence it can summarize the feature of speech signal better.

We will also try some other types of neural works, including Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN). As for the reason, considering that we hope out network "remember" some non-stationary noise patterns, which present lots of regular patterns in spectrum, and CNN is good at recognition of patterns. On the other hand, speech enhancement has great time-relation. Though we have combined frame group of spectrum, its time-relation is still limited, and RNN is good at processing rime-related input data, especially some specific types of RNN, e.g., Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU).

Keywords: Speech Enhancement, Neural Network

目　录

一、 绪 论

通信是人类社会重要的交流方式，而语音消息则是其中之一。而现实世界是一个极其复杂的声学环境，到处都充满了噪声，也带来了很多对降噪技术的需求。人与人之间面对面口头交流其实就是语音消息最常见的一种，根据香农信息论的内容，我们可以大致定性地知道，如果想提高信道容量（可以理解为“听清”说话人说的多少内容），那么在带宽不变（对于面对面交流，带宽可以理解为说话人的语速）的情况下，就必须提高信噪比。相当于如果背景比较安静，说话人可以以较低的音量讲话，而对于比较嘈杂的环境，比如车站、大街、KTV 内，就需要提高音量或者拉近距离才能听清同样的内容。

当然，人类在长期的交流过程中，也会或多或少学会一些“理解”嘈杂语音的方法，一方面可以从所说语言的前后内容上猜测出当前说的内容，一方面人耳与大脑配合也能在一定程度上把背景的嘈杂声音“分离”出来，前者需要对语言有一定的了解，后者需要对噪声的模式（即我们下面将要提到的“特征”）有一个总体上的把握。这样一来，从某种意义上讲，人耳+人脑相当于一个信号系统与神经网络的级联，能够较好的构建从输入的带噪音频到输出（理解）的纯净内容之间的映射关系。

近年来，随着计算机技术的普及与进步，语音消息已经成为人们日常生活中非常重要、常见的消息类型，其需求也在与日俱增。虽然键盘、鼠标、触摸屏仍然是绝大多数电子设备与系统的交互方式，但语音交互已经逐步扩大其所占比例，很多智能设备都拥有了语音交互功能，这个功能对于很多便携智能设备、智能家居、车载智能设备当中显得如虎添翼，解放双手的同时又提高了交互体验。而这些设备通常处在非常复杂的声学环境中，周围有很多随机噪声、突发噪声的干扰，这些噪声会极大影响语音识别设备对语音信号的识别，因此当务之急就是找到有效分离语音与背景噪声的方法。

另外，在实际沟通中语音增强也有用武之地，即使是科技发达的今天，也有很多通信条件极差的信道环境，比如陆空对话，会带有很多飞机的噪声，对于极为重要的空管指令，语音的清晰度则变得性命攸关。因此，防止误解空管指令除了复诵这种传统方法外，还可以加一层语音增强作为双保险。

（一） 语音增强

1. 语音增强的定义

语音增强是指纯净语音在相对较差的声学环境下，受到各种噪声干扰时，使用一定方法将噪声滤除，以提升该段语音的质量和可懂度的技术。如下图 1-1 所示，图 1-1（a）是一段语音在信噪比-5dB 的户外背景噪声下的语谱图（语音采样率 16kHz，因为语音低频成分较多，所以图中只选取 4kHz 以下的部分），图 1-1（b）是纯净语音的语谱图。本论文的主要内容——语音增强的目的就是将左图的带噪语音中分离出右图的纯净语音。从图中可以看出，这种噪声在低频部分能量比较大，里面图案比较复杂，随机性很强，并且存在平稳与非平稳（注意两张图右下角）的噪声成分。

而但从肉眼上对比，也能明显看出右边纯净的语音在左图中的所在部分，所以理论上分离这些成分是完全有可能的。

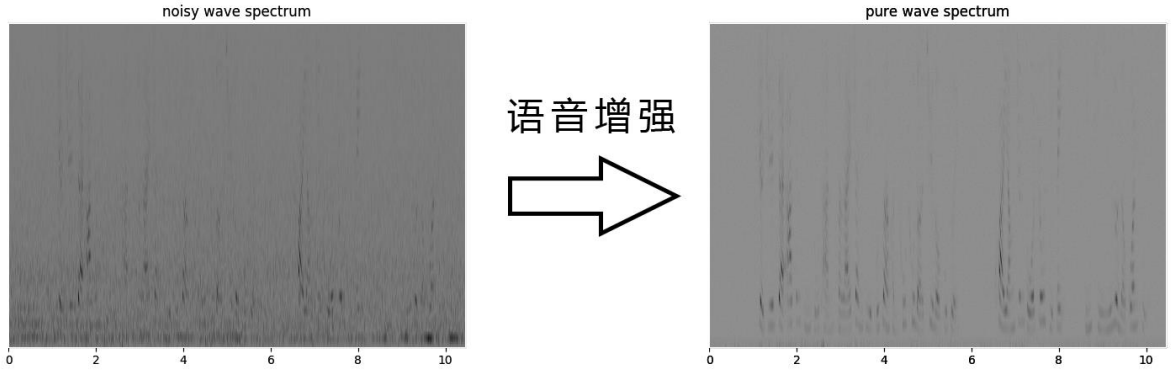

图 1-1（a） 语音增强前　　图 1-1（b）语音增强后

2. 语音增强的分类

语音数据可以按照录音的通道数分为单声道和双声道语音，双声道语音的两个通道相似度非常高，而且我们用于增强的语音很少会有双声道的（或者说，多数双声道语音质量都很好）；另外，虽然人耳的听觉系统虽然是双声道的，可以根据两个声道的不同来辅助对不同声音的分离，但人耳听觉系统即使在听单声道带噪语音也能区分其中的音源。因此，在人耳听觉系统分离带噪语音中有用成分的过程当中，起主导作用的是时域与频域信息，所以本论文将只考虑单声道语音数据。

按照语音增强的方法来分，可以分为无监督语音增强方法和有监督语音增强方法，前者即为传统语音增强方法，它们不需要所谓的“训练”，所需硬件机能不高，但其设计的多数前提在自然界复杂声学环境下的不合理性限制了其性能。而有监督语音增强方法则是最近几年才提出的，原理和人耳听觉系统非常相似，先去学习语音和噪声的

模式，从而学习分离两者的办法。这个“学习”的过程中最主要的部分就是人工神经网络了，本论文也将对此着重研究。

（二） 单声道语音增强方法的发展

与本论文研究的基于神经网络的有监督方法不同，传统的单声道语音增强方法都是无监督的语音增强方法，它们出现非常早，至今已有数十年的历史，大致可分为时域方法和频域方法。时域方法有参数和滤波方法、信号子空间法等。基于参数和滤波方法主要是利用滤波器估计发音器官的声道参数和激励源的激励参数，但通常后者很难估计，尤其辅音的激励与白噪声的随机信号非常相似，而到了低信噪比环境下，这两者变得都难以估计。信号子空间法利用语音信号具有稀疏的特性，把带噪语音信号分解为语音子空间与噪声子空间，消除噪声子空间，保留语音子空间。

频域的方法有谱减法、维纳滤波法等。谱减法原理简单，容易实现，但非常依赖对噪声谱估计的准确性，如果过估计则会引起语音失真，如果噪声欠估计则会产生音乐噪声;维纳滤波法相比而言能较好处理音乐噪声,但其对非平稳噪声抑制能力较弱，且容易造成语音失真。

而近年来方兴未艾的有监督语音增强方法主要有三类:基于人工神经网络的方法、基于隐马尔可夫模型的方法和基于非负矩阵的方法，后两者我们不做详细阐述。基于人工神经网络的方法最初在1994年被提出，但由于当时计算机性能的低下与神经网络技术理论的不成熟，使得该研究并没有深入，更没有得到应用。直到2009年深度学习的概念出现，神经网络理论也相比上世纪大有补充，加之使用神经网络识别语音的方法的成功应用，让基于神经网络的语音增强方法重新焕发光彩，其相关理论我们将在第二章讨论，相关研究将在第三章讨论。

（三） 本论文的研究内容和目标概述

通过前面列举的一些其他语音增强方法的优缺点，我们可以看出我们当前要解决的主要矛盾有哪些。对于无监督的方法，我们希望能够提高对噪声估计的准确性，同时能够针对非平稳噪声有良好的处理结果。对于有监督的方法，我们希望提高网络深度，增加隐藏层节点个数，尽量让网络能够充分拟合带噪语音和纯净语音之间的非线性映射关系；同时提高网络对噪声种类、语音语种、语音性别、信噪比的适应程度，这需要我们使用更加广泛的数据集进行网络训练，数据的种类将直接决定我们的网络在各种条件下的适应程度，即神经网络的泛化能力。

我们的研究目标是搭建神经网络构造一个规模合适、属性好修改、便于我们控制

变量从而定量研究的语音增强器。然后对网络的部分参数进行微小改动，定量分析这些改动对输出语音效果的影响。接着我们将改变网络结构，尝试使用新种类的神经网络来重复上述操作。类似于卷积神经网络的出现对图像识别技术的影响，只使用全连接层的图像识别准确率没有使用卷积神经网络的高，正是因为卷积操作更能概括一幅图像的特征内容。因此在我们的研究过程中，如果结果无法满足要求，我们会考虑改变输入内容，使用更加能够概括一条语音信息，“喂给”神经网络它能更好“理解”的内容，作为网络输入，并判断输出结果的质量改变。

当然，在这些之前，我们还将在第二章里对神经网络做一些简单的入门介绍，并通过数学方法简要证明使用神经网络进行语音增强的可行性。

二、 神经网络初步

人工神经网络（Artificial Neural Network, ANN），简称神经网络（Neural Network, NN）在机器学习领域中是一种模仿动物的中枢神经系统（特别是大脑）的结构和功能的数学模型或计算模型，用于对函数进行估计或近似。神经网络可以通过 “学习”样例的内容，从而完成一系列的任务。例如通过学习一系列标有“猫”、“狗”等标签的图像样例后，对未见的猫或狗图像进行识别。大多数情况下人工神经网络能在训练样例的基础上改变其内部结构，是一种典型的“自适应系统”。

（一） 神经网络的基本结构

1. 神经元

神经网络中有大量的人工神经元（生物大脑中神经元的简化版）相互联结，部分神经元之间可以相互传输数据，从而进行计算任务。

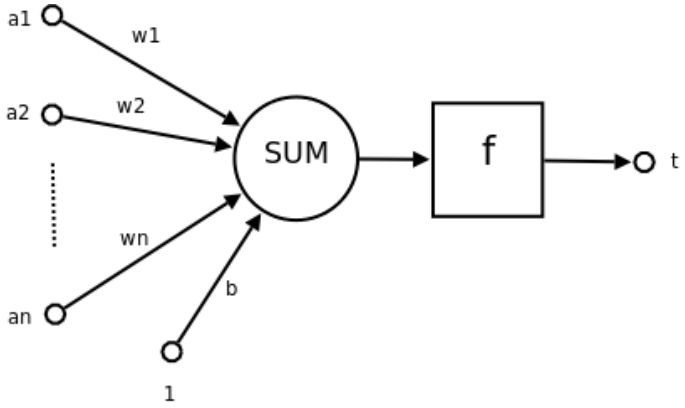

图 2-1 神经元的基本结构

神经元中最常见的参数有：

1) 输入向量（图 2-1 中$\vec{A} = [a_1, a_2, \dots, a_n]^T$）
2) 输入权值向量（图 2-1 中$\overrightarrow{W} = [w_1, w_2, \dots, w_n]^T$）
3) 偏置值（图 2-1 中b）
4) 激活函数（activation function, 图 2-1 中f）
5) 输出值（图 2-1 中t）

一个神经元的输入端通常与其他多个神经元的输出端相连，每个神经元的输出是一个数值a，而多个神经元的输出合并在一起就是作为下一个神经元的输入，通常我们

把这些输入值写成列向量，称为“输入向量”。神经元还会有一个自带的“权值向量”和“偏置值”，权值向量和输入向量的点积加上偏置值后，其和作为神经元的激活函数的输入，激活函数通常来说是非线性的，这样才能更好的拟合各种非线性的映射，这点我们将在以后讨论。最后，函数的输出值就是整个神经元的输出。于是我们有

$$t = f\left(\vec{W} \cdot \vec{A} + b\right). \qquad (2\text{-}1)$$

可见，神经元的所要做的是把$\mathbb{R}^n$空间中的一个矢量$\vec{A}$映射到$\mathbb{R}$上。

2. 神经网络

有了对神经元概念的基本了解，我们接下来引入神经网络的概念。神经网络按照层数可以分为单层神经网络和多层神经网络，如图 2-2 所示。图 2-2（a）是一个三输入、三输出的单层神经网络，假设三个输入量写成向量形式为$\vec{X}$，则输出向量$\vec{Y}$就是

$$\vec{Y} = f\left(\begin{bmatrix}\vec{W}_1\\ \vec{W}_2\\ \vec{W}_3\end{bmatrix} \cdot \vec{X} + \begin{bmatrix}b_1\\ b_2\\ b_3\end{bmatrix}\right). \qquad (2\text{-}2)$$

而图 2-2（b）的多层神经网络是一个三输入，二输出，其原理与单层神经网络类似，其中左边三个为输入层（Input Layer），右边两个为输出层（Output Layer），中间四个为隐藏层（Hidden layer），简称“隐层”。隐藏层用于连接输出层和输出层，可以有多层，习惯上会至少有一个，其节点数不定，但可以肯定的是，隐藏层神经元数目越多，神经网络的非线性就越显著，从而提高神经网络的鲁棒性（Robusteness）。由于单层神经网络性能不强，拟合能力较差，现在已经不常使用，因此本论文接下来将只讨论多层神经网络。

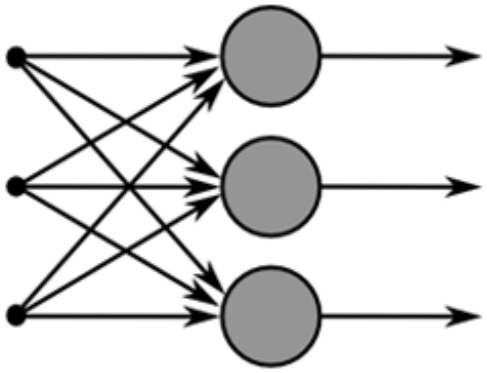

图 2-2 （a）单层神经网络

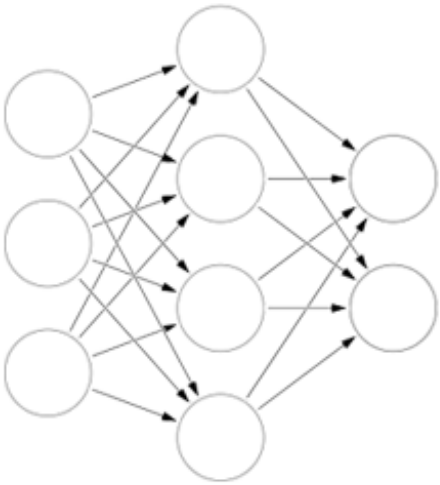

图 2-2 （b）多层神经网络

（二） 神经网络的训练方法

神经网络的学习过程，是通过分析训练样本输出值的差异，来矫正网络中权重值与偏置值的过程。首先我们要设定一种分析输出值的方法，通常我们会定义一个代价函数（Cost Function），也叫误差函数（Loss Function）。假如我们设定代价函数为网络输出与实际值的平方的均值（即均方误差，Mean Square Error, MSE），那么我们就能得到一个以权重值和偏置值为自变量、均方误差为因变量的函数，假设权重值与偏置值有n个，那么神经网络的训练方法可以理解为寻找一个定义在$\mathbb{R}^n$上的函数$g: \mathbb{R}^n \to \mathbb{R}$的极小值的过程。下面我们将讨论寻找函数$g$的极小值过程中的具体细节。

1. 梯度下降法

梯度下降法（Gradient descent）是一个一阶最优化算法，通常也成为最速下降法。方法是从函数当前点对应梯度的反方向规定步长（在机器学习里，步长常称为“学习率”）点进行迭代搜索。

考虑一个n元实值函数$g: \mathbb{R}^n \to \mathbb{R}$在$\mathbb{R}^n$中定义域内一点$a$处$C^1$，则函数$g$在点$a$处沿着梯度的反方向$-\nabla g(a)$下降最快。令

$$\mathrm{b} = \mathrm{a} - \gamma \nabla \mathrm{g(a)} \tag{2-3}$$

若能找到一个足够小的数γ，使得

$$g(a) \geq g(b) \tag{2-4}$$

并以此迭代，从初始值x_0出发，考虑如下序列

$$x_{n+1} = x_n - \gamma_n \nabla g(x_n) \tag{2-5}$$

可以得到

$$g(x_0) \geq g(x_1) \geq g(x_2) \geq \cdots, \tag{2-6}$$

那么序列$\{x_n\}$就能收敛到期望的极值点，要注意的是，式（2-5）中的学习率γ_n可以改变。

但是，梯度下降法也有一定的不足，首先从式（2-5）就能看出，对于函数比较“平缓”的部分，梯度值$\nabla g(x_n)$很小，那么每次迭代的步长$\gamma_n \nabla g(x_n)$也会很小，这会造成序列$\{x_n\}$在平缓处接近极小值的速度很小。在机器学习里，就会出现代价函数降低缓慢，从而训练效率低下的现象。第二，对于在极小值附近出现非常“狭长”的山谷形状的函数，梯度下降法可能导致每次迭代，$\{x_n\}$走一个之字形路线，这同样会引起训练效率低下。

2. 反向传播算法

反向传播算法（Backpropagation，BP）全称“误差反向传播”，通常与梯度下降法等最优化方法结合使用，是一种训练神经网络的常见方法。该方法可以对网络中所有权重、偏置值计算代价函数的梯度，而这个梯度会反馈给最优化方法，从而更新权重、偏置值来最小化代价函数。

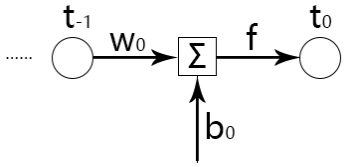

图 2-3 误差反向传播的解释

为了解释反向传播算法的具体原理，我们只取神经网络的两个神经元，如图 2-3，t_0是神经元的输出值，t_{-1}是上一个神经元的输出值。我们假设t_0即为神经网络的输出，并且真实值应当为y，另设代价函数c为均方误差

$$c = \frac{1}{2}(y - t_0)^2. \tag{2-7}$$

注意，我们在这里加了一个常系数$\frac{1}{2}$，这个系数选哪个常数并没有关系，而我们选$\frac{1}{2}$是为了稍后计算方便。同时，根据式（2-1），我们有

$$t_0 = f(t_{-1} \cdot w_0 + b_0). \tag{2-8}$$

由于代价函数c直接与权重、偏置值有关，所以我们想求其对这两者的偏导数，于是代价函数对权重的偏导数有

$$\frac{\partial c}{\partial w_0} = \frac{\partial c}{\partial t_0} \cdot \frac{\partial t_0}{\partial w_0} = (t_0 - y)(f' \cdot t_{-1}), \tag{2-9}$$

代价函数对偏置值的偏导数有

$$\frac{\partial c}{\partial b_0} = \frac{\partial c}{\partial t_0} \cdot \frac{\partial t_0}{\partial b_0} = (t_0 - y)(f'). \tag{2-10}$$

我们可以发现，这两者的计算用到了微积分学里的“链式法则”，注意到t_{-1}又是上一层神经元的输出，那么代价函数对上一层神经元的权重、偏置值的偏导数同样可以继续应用链式法则来计算。而对于多个神经元的多层网络，原理也是一样的。

当我们利用链式法则计算出代价函数对每一个权重、偏置值的偏导数之后，我们要清楚这些偏导数的值的实际含义——偏导数的绝对值越大，说明自变量的微小变化带来的因变量的变化就越大，即因变量对那个自变量就更敏感。于是我们就有了通过梯度下降法配合反向传播算法调整所有权重、偏置值的方法：先计算代价函数的负梯度，这是梯度下降法的部分；然后通过链式法则求出所有自变量的偏导数，然后将负梯度乘以学习率得到的步长应用到所有自变量的改变上，从而让网络得以修正、学习。而链式法则是从网络输出端开始反向计算的，因此该算法得名“反向传播算法”。

（三） 神经网络的函数逼近性能

前面我们了解了神经网络的基本结构与训练方法，接下来我们要通过数学方法阐述神经网络对于函数的逼近性能，以此作为我们接下来将神经网络用于语音增强的理论依据[6]。由于篇幅有限，下面的数学定理只作阐述，不作进一步证明。

定理 1：设φ为有界单调递增连续函数，$K\subset \mathbb{R}^n$为紧集，固定层数$k=3$，则对$\forall f: K \mapsto \mathbb{R}^n$可被一$k$层（$k-2$层隐藏层）网络一致逼近，其中$f$是$C^0$的。

这一定理中，φ即为我们之前提到过的激活函数。可以发现，只需要很少层（定理中是三层网络，包括一个输入层，一个输出层和一个隐藏层）的神经网络就能一致非常复杂的函数。

定理 2：$\forall \varepsilon > 0, f \in L^2[a,b]$，∃一个三层神经网络，使逼近误差按$L^2$范数小于ε.

该逼近定理证明了三层神经网络对二次可积函数无限逼近的存在性，但并没有给出具体这种网络的构造准则，而接下来的定理 4 则能让我们了解神经网络隐藏层激活函数的选择方法。

定义 1：若函数$\varphi: \mathbb{R}^1 \mapsto \mathbb{R}^1$满足：

$$\varphi(x) \to 0, x \to -\infty,$$
$$\varphi(x) \to 1, x \to +\infty,$$

则称φ为 Sigmoid 型函数。

定义 2：（Tauber-Wienner 函数）如果函数$f: \mathbb{R}^1 \mapsto \mathbb{R}^1$（连续或不连续）满足：所有如下的线性组合

$$\sum_{i=1}^{N} c_i f(\lambda_i x + \theta_i), \lambda_i, \theta_i, c_i \in \mathbb{R}$$

在$C[a,b]$中稠密，则称f是一个 Tauber-Wienner 函数，其全体记为(TW).

定理 3：如果σ是一个有界 Sigmoid 型函数，则$\sigma \in (TW)$.

定理 4：设 $K\subset \mathbb{R}^n$为紧集，$f \in C(K)$，$\sigma \in (TW)$，则$\forall \varepsilon > 0, \exists N > 0, \theta_i \in \mathbb{R}, w_i \in \mathbb{R}^n$与常数$c_i(f), i = 1,2,\dots,N$，使得

$$\left| f(x) - \sum_{i=0}^{N} c_i(f)\sigma(w_i x + \theta_i) \right| < \varepsilon.$$

定理 4 告诉了我们，一个函数$\sigma \in (TW)$是其能作为神经网络隐层激活函数的充要条件，从而大大缩小了我们构造神经网络时对激活函数的选择范围。

然而，刚刚我们说的只是在保证无限逼近某函数的条件下对激活函数的要求，在实际当中，我们是允许神经网络的最终逼近性能有些许误差的，而我们更在意的是网络的训练速度快慢，以及训练这个网络对我们硬件要求的高低。也就是说，如果选择一些(TW)之外的函数，比如机器学习领域常用的线性整流函数 ReLU，不但可以让我们神经网络逼近一个我们能接受的误差，还能提升训练速度，降低训练的硬件需求。

不过值得欣慰的是，上面的证明至少给我们指明了一个明确的方向，即假设带噪语音信号的特征与纯净语音信号的特征之间的映射是连续的 n 元实值函数，那么通过神经网络来增强语音是可行的。

三、 基于神经网络的语音增强方法

在前一部分里，我们介绍了神经网络的基本结构与训练方法，并通过数学方法阐述神经网络对于函数的逼近性能，得知通过神经网络来增强语音是可行的。在这一部分，我们将利用这一结论，设法构造网络并训练之，最终得到一个能够增强语音的神经网络。

首先，对于所有神经网络的训练，都需要先有足够多的训练集支撑。我们将收集大量的语音数据与噪声数据，并利用 MATLAB 软件，合成大量的、不同信噪比的噪声数据，作为我们网络的训练集。

其次，我们将尝试搭建网络，首先尝试的是深度神经网络，这是结构最简单，也是最常用一种网络类型。网络的输入与输出将使用语音信号的短时傅里叶变换取对数再高斯归一化，然后将其加帧组，作为时间关联性的补充，最后输出单帧的数据，并反向上述操作。

最后，我们将尝试改进神经网络，将语音的梅尔倒频谱（Mel-Frequency Cipstal Coefficients, MFCC）加入到网络输入层，研究语音的改进情况。

另外，值得注意的是，上述全部内容都是对语音信号的幅度操作，而我们可以从主观角度证明，人耳对声音的相位并不敏感，因此，最终合成的语音信号的相位将使用带噪语音的相位。

（一） 数据集的生成

为了生成不同信噪比的带噪语音数据，我们首先收集了一万余条汉语语音数据，平均每条十秒，与 20 种 QUT-NOISE-TIMIT 数据集中的噪声数据，总长度超过 13 小时。然后利用 MATLAB 软件，生成-10dB，-5dB，0dB，5dB，10dB 信噪比的，每种信噪比 20000 条的带噪语音数据。

首先固定第一条噪声数据，从一万余条汉语语音数据中随机选择 1000 条，对于每条数据，我们从噪声数据中随机截取一段来，按照所需的信噪比调整幅度后相叠加。重复上述操作至所有 1000 条语音数据，再使用第二条噪声数据重复上述操作，直至所有噪声数据都被用上。这样我们就有了$20 \times 1000 = 20000$条带噪语音数据。

需要注意的是按照信噪比结合的方法，设纯净语音信号为$x(n)$，噪声信号为$g(n)$，信噪比为SNR，则最终生成的带噪语音信号$y(n)$为

$$y(n) = x(n) + g(n) \times \frac{\|x(n)\|_2}{\|g(n)\|_2} \times 10^{\frac{SNR}{20}}, \qquad (3\text{-}1)$$

其中$\|x(n)\|_2$指向量$x(n)$的二范数（因为本论文提到的所有语音信号都是单声道信号，所以$x(n)$可以当成一个向量），其定义为

$$\|x\|_2 = \sqrt{\sum_{i=1}^{N} x_i^2}. \quad (3\text{-}2)$$

选择这样的方式生成训练集，一方面好处是生成的训练集数量可以非常大，即使我们最开始只有一万条左右的纯净语音数据，利用这种方法我们依然生成了两万条带噪语音数据；另外，随机的选取也可以保证训练集的泛化程度高，不会在训练的时候使神经网络因反复学习某种单一的语句样式从而陷入局部最优解，有利于神经网络在学习过程中结果更加收敛于全局最优解。

（二） 基于深度神经网络的语音增强

1. 工具的选择

首先是编程语言的选择，近年来，Python 因其简单易上手，又有海量强大的库而风靡全球。作为一种面向对象的解释型语言，Python 虽然经历过一次非常痛苦的版本变革，但如今 Python3 已经拥有非常多库的支持，其中一些科学计算的库，比如 numpy，更是拥有与 MATLAB 匹敌的矩阵计算速度。

其次是 Python 具体库的选择。TensorFlow 是一个谷歌的开源机器学习库，最开始是由谷歌大脑（Google Brain）团队开发，用于谷歌内部的使用，后来于 2015 年 11 月 9 日在 Apache 2.0 开源许可证下发布。TensorFlow 支持的编程语言有 Python 和 C++，支持的系统平台有 Linux、macOS 和 Windows，支持 CPU 与 GPU 运算。TensorFlow 使用起来略微复杂，于是我们还需要 Keras 库作为辅助。

Keras 是一个用 Python 编写的高级神经网络 API，它能以 TensorFlow、CNTK 或 Theano 作为后端运行。它最大的有点是用户友好、高度模块化，能够极大提高神经网络的开发效率，并且支持 GPU 运算。对于想把它作为“黑箱子”使用的人，Keras 是一个非常不错的选择。

本文将使用 Python 版本 3.6 与 TensorFlow-GPU 版本 1.7.0 与 Keras 版本 2.1.5，同时使用 numpy 版本 1.14.2 作为辅助计算软件，对于音频的处理，我们还需要安装 scipy 版本 1.0.1 用于短时傅里叶变换与其逆变换，python-speech-features 版本 0.6 用于语音数据梅尔倒频谱的生成，以及 matplotlib 版本 2.0.2 用于语谱图的绘制。以上库在装好 Python 后均可以用 Windows 命令行中的“pip install XXX”命令来安装。需要注意的是，TensorFlow 虽然支持 CPU 与 GPU 运算，但在安装的时候最好只安装一个版本的，两个版本同时存在可能会出现一些无法预知的错误。至于硬件，我们的 CPU 将使用英特尔 i7 4770K，GPU 将使用英伟达 GTX780Ti。前者主要用于输入、输出数据的加工处

理（例如短时傅里叶变换、添加帧组、归一化等操作），后者则主要参与神经网络训练的运算工作。当使用 GPU 作为 TensorFlow 的硬件时，应当安装其适配版本的英伟达 CUDA 驱动，可以在英伟达的官网找到，安装好后还要下载 cuDNN 组件，这是用 TensorFlow 做 GPU 运算的必备组件。这样我们的准备工作就完成了。

2. 输入、输出数据的处理

首先我们要构建输入数据。因为我们是基于神经网络的语音增强研究，所以我们会把训练集数量作为研究的内容，因此就涉及到了训练集选取上，前面我们已经生成了 20000 条语音数据，而这些我们不会全部使用，我们将只选取其中 900 条到 7200 条不等。而选取的方法就显得很重要，因为作为训练使用，我们必须同时拥有带噪语音和纯净语音，而这两种语音是分别存放在两个文件夹内的。如果我们各自从两个文件夹内随机取某个数量的语音数据，最后结果就是纯净与带噪语音完全不是一条，这是十分荒唐的。所以我们需要先将两种数据两两“绑定”起来，然后一对一对地随机选取。还好 Python 的 sklearn 库中提供了名叫“shuffle”的方法，可以让我们实现上述操作。另外我们还把这个操作包装在一个方法中，并定义一个控制选取文件量的参数，这在我们将来定量分析时会提供很大便利。

接着，对于每条语音数据，我们取语音信号$x(n)$的短时傅里叶变换

$$X(l,k)=\sum_{n=-N/2}^{N/2-1} w(n)\cdot x(n+lH)e^{(-j2\pi kn/N)}, \qquad (3\text{-}3)$$

其中l指第l帧，$w(n)$是窗函数，H是步长，默认值为窗函数宽度的一半。短时傅里叶变换在 Python 的 scipy 库中已有提供，本论文将使用其默认参数，其中窗函数选用汉宁窗：

$$w(n)=\frac{1}{2}\left(1-cos\left(\frac{2\pi n}{N-1}\right)\right), \qquad (3\text{-}4)$$

其时域图像与频域图像如图 3-1 所示。窗函数宽度为 256 点，步长 128 点。我们使用的语音信号采样率是 16kHz，那么对于t秒的数据，一共有$16000t$个采样点，那么帧数 L 则为$16000t\ /\ 128-1$，每帧 129 个数据。

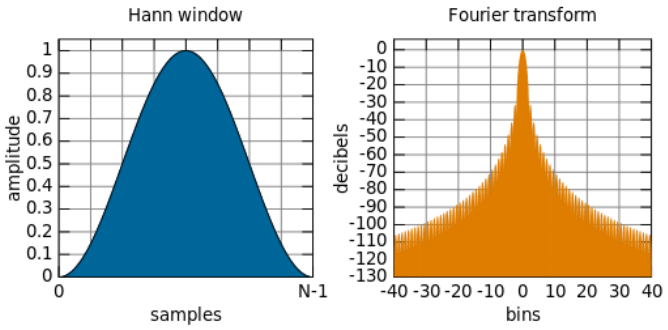

图 3-1 汉宁窗的时域图像与频域图像

之后我们将把这些数据取模再取对数，构成对数功率谱，其定义为

$$Y(l,k) = log|X(l,k)|^2. \quad (3\text{-}5)$$

接着，我们将$Y(l,k)$再进行高斯归一化，即整个训练数据的均值被规整到 0，方差被规整到 1，公式如下：

$$Z_{normalized} = \frac{Y-\bar{Y}}{std(Y)}. \quad (3\text{-}6)$$

其中$\bar{Y}$指Y的均值，$std(Y)$指Y的标准差。

这样处理的好处是将网络的输入数据（稍后用于监督的输出数据我们也将同样处理）统一规整到同样的范围内，这是非常有利于网络的学习的。在第二章内我们提到过，梯度下降法的一个缺点是对于非常“狭长”的山谷形状的函数，梯度下降法可能导致每次迭代会呈现“之”字形路线，从而影响网络学习的速度。而对数据提前进行高斯归一化则恰能避免这样的问题，化“狭长”的函数变为规整形状的函数。

经过上述操作，我们得到了一组均值为 0、方差为 1 的，频率轴 129 维，时间轴有$16000t \,/\, 128 - 1$帧的数据，下一步我们要加帧组。首先要说明加帧组的目的，我们知道语音信号本身是有很强的前后关联性的，由于单帧信号时间长度只有$256 \div 16000 = 32$毫秒，显然在时间上的关联性不足。如果我们只以单帧作为网络输入，则很难得到非常好的语音增强效果。于是，如图 3-2 所示，我们将每一帧（中心帧）旁边的多帧数据组合成帧组，一同作为网络的输出。同时，下一帧组是上一帧组向后移动一帧后的数据。在程序中，我们会定义一个整型变量叫“neighbor”，意义为每一帧与其两边各多少帧组合成帧组，例如当 neighbor 等于 4 时，每个帧组里就有$4 + 1 + 4 = 9$帧的数据，这样输入层就有$9 \times 129 = 1161$个数据，neighbor 等于其他值时同理。另外，对于一段语音数据两端的部分，由于缺少一侧的 neighbor，我们将重复其中心帧本身作为 neighbor。这样我们的输入数据就准备完成了。

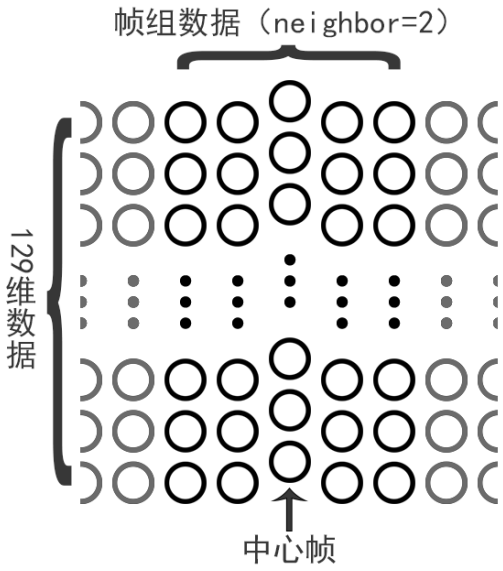

图 3-2 帧组数据说明

而对于输出数据，处理则简单得多，我们将只使用输入帧组中心帧的数据作为输出。而在网络学习成熟之后的应用中，对于输出的每一帧数据，我们将把它组合回$129 \times (16000t / 128 - 1)$维的数组，再将其反归一化，反归一化的过程就是式（3-6）的逆运算

$$Z_{unnormalized} = Z_{output} \times std(Y) + \bar{Y}. \quad （3\text{-}7）$$

这里我们将使用带噪语音数据的$\bar{Y}$和$std(Y)$。再取其e指数作为反变换的模值，将它与带噪语音语谱的相位值相组合，共同作为短时傅里叶反变换的输入。数学形式如下

$$X_{output} = exp(Z_{unnormalized}/2) \cdot exp[j \times arg(X)]. \quad （3\text{-}8）$$

注意，式（3-8）中的乘法不是矩阵乘法，而是矩阵每一项的算术乘法。最后，计算X_{output}每一帧的短时傅里叶反变换

$$x(l) = \frac{1}{L}\sum_{k=0}^{L-1} X(l,k)e^{j2\pi kl/L}, \quad （3\text{-}9）$$

再使用重叠相加算法合成整条语音数据。

最后还要说明的是，在语音数据的保存上，Python 与 MATLAB 略有区别，需要手动将输出的时域数据转换成 int16 类型再进行保存，因为其 scipy 自带的 istft 方法的返回值类型是 double64，直接保存声音会非常刺耳。

3. 神经网络的搭建

我们使用 Keras 来搭建神经网络，首先需要定义一个 Sequential 对象 model 作为神经网络的初始化，然后每增加一层网络就要调用一次 add 方法，增加一个 Dense 对象。Dense 对象的具体参数有神经元个数（若为输出层，这个参数则代表输出层数据，比如当前我们输出层数据有 129 个）、激活函数名、数据类型，如果是输入层，还要定义输入的大小。

```
model = Sequential()
# input layer to hidden layer 1   frame * framewidth float -> hidden neurons float
model.add(Dense(hid_neus, activation='relu', dtype=tf.float32, input_dim=frwd*fr))
# hidden layer 2   hidden neurons float -> hidden neurons float
model.add(Dense(hid_neus, activation='relu', dtype=tf.float32))
# hidden layer 3   hidden neurons float -> hidden neurons float
model.add(Dense(hid_neus, activation='relu', dtype=tf.float32))
# output layer   hidden neurons float -> framewidth float
model.add(Dense(frwd, activation='linear', dtype=tf.float32))
```

图 3-3 定义神经网络结构部分的代码

如图 3-3 展示了详细代码，我们在代码之前定义了 hid_neus 变量为 2048，这代表每层隐藏层含有的神经元数量，需要注意的是，该神经元数量最好要大于输入层的个数，因为我们希望神经网络能够学习输入数据，即帧组的全部特征信息，如果神经元

数量太少，则相当于对信息进行了压缩，这是我们不希望看到的。同时我们还定义了 frwd 变量，是每一帧内的数据数，即 129，以及 fr 变量，是每个帧组所含帧数，这样两者相乘即为输入数据的维度。

我们使用线性整流函数（ReLU）隐藏层的激活函数，其数学定义如下

$$ReLU(x) = max(0, x). \quad (3\text{-}10)$$

关于使用 ReLU 作为激活函数的优点，目前普遍认同的解释有三[5]：

1) 不会出现梯度消失。因为 ReLU 函数在激活的时候梯度恒等于 1，与之相比，无论是 sigmoid 函数还是 tanh 函数，它们在激活的时候，随着自变量的增大，梯度都会趋近于 0，使其梯度（或者说前文提到的，反向传播算法中用到的偏导数）会趋于 0，即出现“梯度消失”的现象；
2) 稀疏性（Sparsity）好。神经网络的稀疏性可以理解为其对特征的概括能力，当使用最少的、最准确的特征作为依据，就能有更优秀的训练效果。另外，ReLU 的特性也更加接近生物神经网络的激活方式；
3) 计算量小——相比于 sigmoid 函数或 tanh 函数来说，这个特点是毋庸置疑的，而相比前两点，这点也是最不重要的。

然后使用线性函数（即$y = x$）作为输出层的激活函数，这是很多进行类似任务的神经网络都会使用的输出层激活函数。

接着，我们定义代价函数，本论文将使用之前提到过的均方误差作为代价函数，其数学定义为

$$MSE = \frac{(y_{predict} - y_{real})^2}{N}. \quad (3\text{-}11)$$

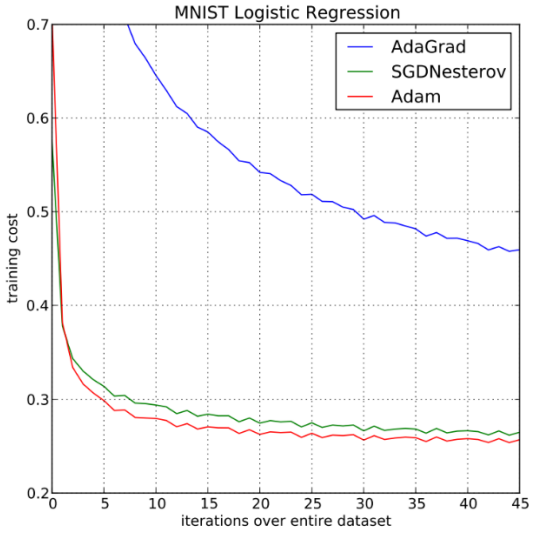

图 3-4 Adam 优化器性能对比

我们使用 Adam 优化器对均方误差进行优化，并设置学习率为 0.00002。Adam 优化器利用梯度的一阶矩估计和二阶矩估计来动态调整没个参数的学习率，优点是计算效率高、内存占用小，尤其适用于高维空间与大数据集的优化任务，且非常善于处理

非平稳目标，可见 Adam 优化器简直就是为本论文的任务目标量身定制的。图 3-4 展示了 Adam 优化器相比于其他优化器的性能。

最后，输入数据与用于监督的输出数据我们在上面已经准备好，我们只需调用 model 实例的 fit 方法来训练网络即可。

4. 语音质量衡量指标

PESQ（Perceptual Evaluation of Speech Quality）是语音质量自动评估的一组方法，曾广泛用于电话通信领域，其标准化形式记录在国际电信联盟电信标准化部门（International Telecommunication Union Telecommunication Standardization Sector, ITU-T）建议书 P.862 上。最初，PESQ 被设计为专门用于远程通信的人类听觉主观测试模型，于是，PESQ 需要真实的语音样本作为测试信号。而如今，PESQ 已成为世界广泛接受的语音质量主观评判的工业标准，应用于电话制造商、网络设备提供商与通信产业等诸多领域。

本论文也将使用 PESQ 作为网络输出结果的衡量标准，在 ITU 的官方网站上我们可以下载到其建议书 P.862 上的 C 语言代码，编译成可执行文件，就可以进行语音评估了。为了能够方便地大量进行评估，我们还为此编写了 Python 与 MATLAB 代码，前者用于让网络批量输出一定数目的强化后语音，后者调用刚刚编译后的可执行文件，批量评估这批强化后语音的 PESQ 评估值。

（三） 基于深度神经网络的语音增强结果评估

在上一部分，我们搜集并生成了数据集，选择了合适的机器学习工具，搭建了神经网络，并选择了合适的语音评估标准。同时我们也初步训练了网络，并取得一些效果。图 3-5 为带噪语音的时域图，图 3-6 为网络输出语音的时域图，图 3-7（a）为带噪语音的语谱图，图 3-7（b）为网络输出语音的语谱图。在这一部分，我们将改变其中一些参数，利用控制变量法，分析这些参数对结论的影响。

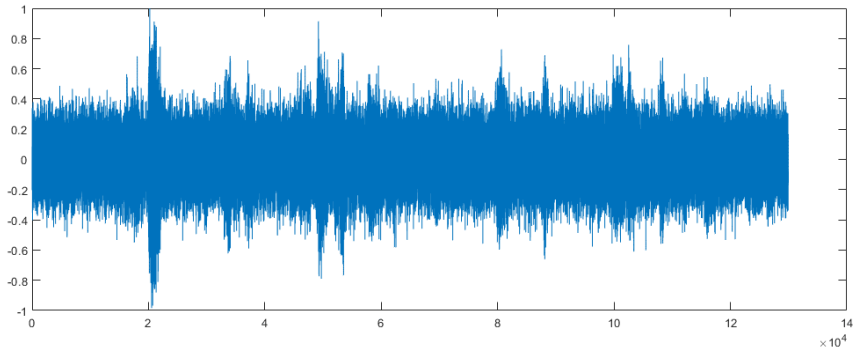

图 3-5 带噪语音时域图

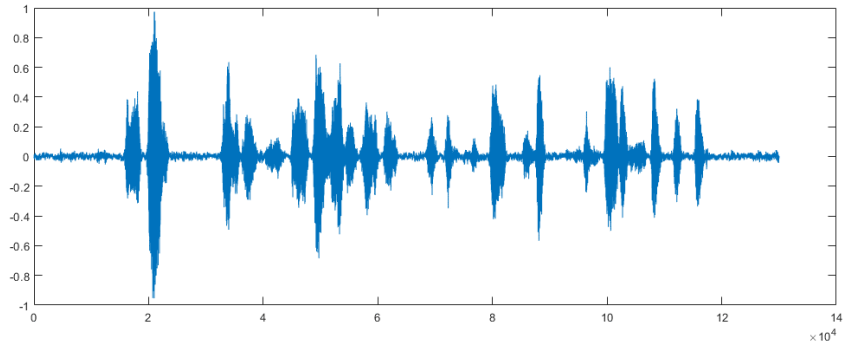

图 3-6 网络输出语音时域图

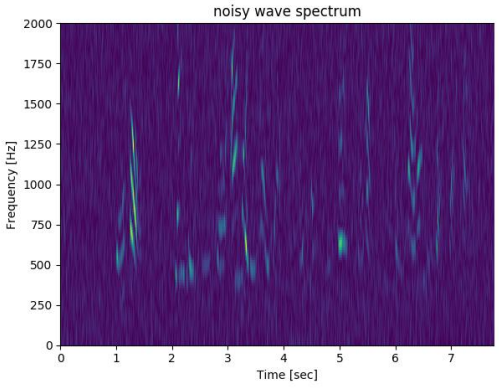

图 3-7（a）带噪语音语谱图

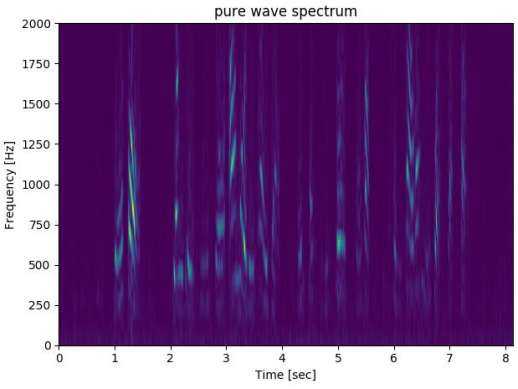

图 3-7（b）网络输出语音语谱图

1. 信噪比的影响

首先我们统计的是不同信噪比下神经网络输出结果的 PESQ 值。我们在本章节第一部分里，分别生成了-10dB，-5dB，0dB，5dB，10dB 信噪比的数据，并各自训练了五个神经网络模型。模型具体细节为：帧组内帧数，9 帧；隐藏层数量，3 层；训练用语音条数，3600 条（总长约 10 小时）。最后，网络输出 100 条随机语音数据，并评判每条语音的 PESQ 值，取这 100 个 PESQ 值的平均数，得到以下图 3-5 与表 3-1。

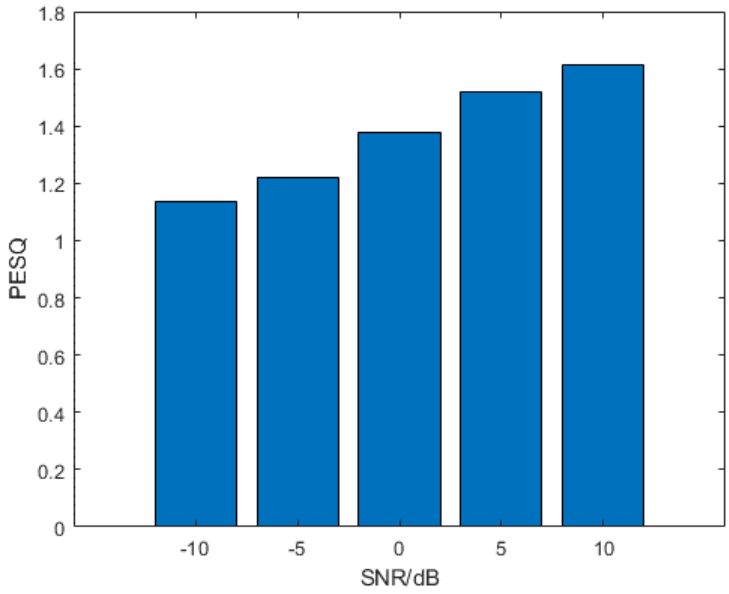

图 3-8 不同信噪比下神经网络输出结果的 PESQ 值柱形图

表 3-1 不同信噪比下神经网络输出结果的 PESQ 值统计表

输入 SNR/dB	-10	-5	0	5	10
PESQ	1.1351	1.2218	1.3788	1.5192	1.6140

从图 3-5 我们可以发现，信噪比越高的带噪语音，神经网络对其增强效果就越好，这是很好理解的。

2. 帧组内帧数的影响

接下来，我们将只以-5dB 噪声为例，调整神经网络的参数，观察输出结果的 PESQ 值。首先修改的是帧组内帧数，我们分别搭建了 1 帧、5 帧、9 帧、11 帧、13 帧为输入的模型，并分别训练之（其中 1 帧作为帧组即相当于不加帧组），其余参数为训练用语音条数，3600 条（总长约 10 小时）；隐藏层数量，3 层；信噪比-5dB，并得到了如图 3-6 与表 3-2 的结果。

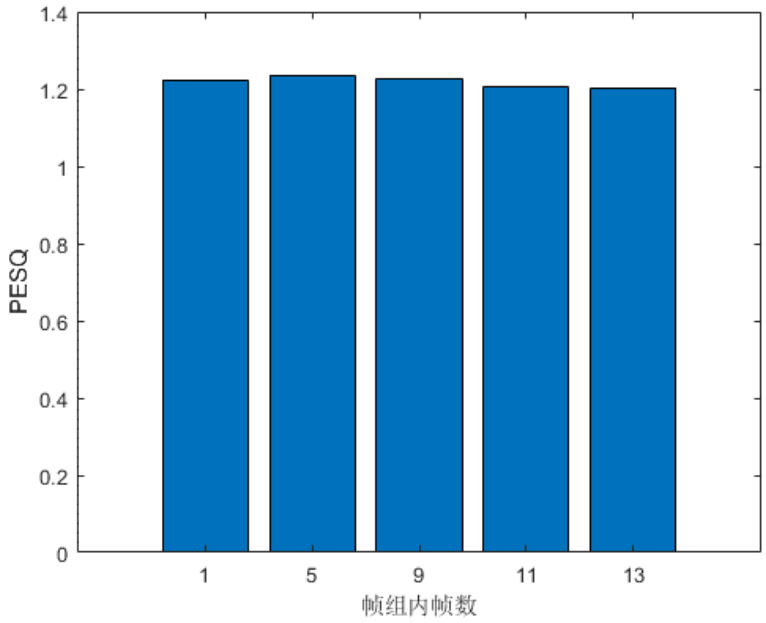

图 3-9 不同帧组内帧数下神经网络输出结果的 PESQ 值柱形图

表 3-2 不同帧组内帧数下神经网络输出结果的 PESQ 值统计表

帧数	1	5	9	11	13
PESQ	1.2239	1.2361	1.2250	1.2050	1.2035

反直觉的是，数据显示帧组内帧数量在 5 的时候会达到一个最大值，比这个值大或小，都会让最终输出结果降低。比这个值小的，即帧数为 1 的时候，PESQ 低这点较好理解，因为神经网络的输入没有时间关联性，所以神经网络无法完全学习语音数据的特征。但当帧数较多时，PESQ 依然会降低，我们认为有两种可能的原因：

1) 神经网络输入量过大，包含特征过多，且冗余较大，神经网络无法在仅有 3600 条训练集的情况下完成对全部特征的映射的学习，即“欠拟合”；
2) 另一种可能是，神经网络陷入了局部最优点，即“过拟合”。

两种情况发生的原因可以说是完全相对的，但造成的结果都是现在这个“反直觉”的数据结论。欠拟合与过拟合也是机器学习中非常棘手、急需解决的两大问题。前者可以单纯的靠增加训练集来实现，不过需要注意的是，在某些情况，比如训练集非常少且难以重现，比如大型昂贵机器的故障，解决办法就不是那么容易想到了，也因此催生了最近“少样本学习”（One-shot Learning, Zero-shot Learning, 有时候也叫 Few-shot Learning）的理论进展，连著名围棋 AI，AlphaGo 的所在公司 Deep Mind 也对此发表过论文[3]。而过拟合则是机器学习里更加需要注意的问题，解决方法也有很多种，常用的一种叫“丢弃法”（Dropout），具体是在训练过程中随机隐藏部分隐藏层神经元，其余神经元保持正常训练过程修改其参数，被隐藏的则不修改，下次训练时重新随机选择需要隐藏的神经元。由于神经网络的非线性，丢弃法具体理论并不完全，但直观上看，整个丢弃的过程就相当于对很多歌不同的神经网络取平均，不同的神经网络会有不同的过拟合表现，而一些互为“反向”的过拟合则可以通过取平均相互抵消，从而整

体上减少过拟合的出现。

3. 训练集数量的影响

接下来我们看训练集数量对结果的影响，我们用同样的一个初始神经网络，分别取 900 个、1800 个、3600 个、7200 个训练集样本进行训练，其余参数为帧组内帧数，9 帧；隐藏层数量，3 层；信噪比-5dB。数据如图 3-7 与表 3-3。

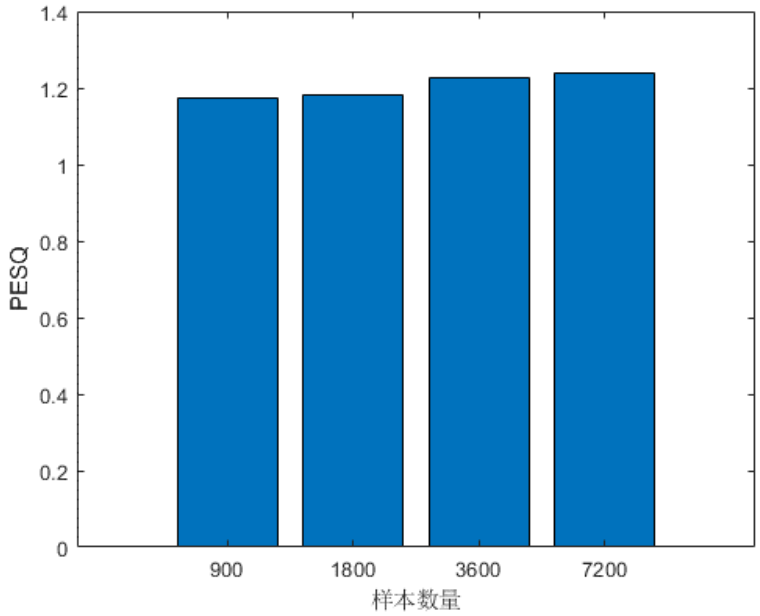

图 3-10 不同训练集样本数量下神经网络输出结果的 PESQ 值柱形图

表 3-3 不同训练集样本数量下神经网络输出结果的 PESQ 值统计表

训练集数量	900	1800	3600	7200
PESQ	1.1725	1.1799	1.2250	1.2405

可以看出，随着训练集样本数量的上升，神经网络能输出更高质量的语音。因此，可以粗浅地认为，如果我们有更多数量、更多种语言、口音、音色的语音数据，与更广泛的噪声类型，我们的神经网络就能够获得更强的性能。

4. 神经网络层数的影响

最后，我们将探究神经网络层数对 PESQ 值的影响，这部分的研究可以让我们看到，对于语音信号的增强任务，多深的网络是合适的。因为在有些时候，任务并不需要太多深度的网络即可完成，此时再继续增加神经网络结构，不会得到更加优异的结果，却会提高计算的复杂度。所以我们的目的是找出此任务最合适的网络结构，减少训练时间，未来应用时的硬件开销。因此，我们将取神经网络隐藏层的数量为 1，2，

3，4 层，其余参数为帧组内帧数，9 帧；训练用语音条数，3600 条（总长约 10 小时）；信噪比-5dB。数据如图 3-8 与表 3-4。

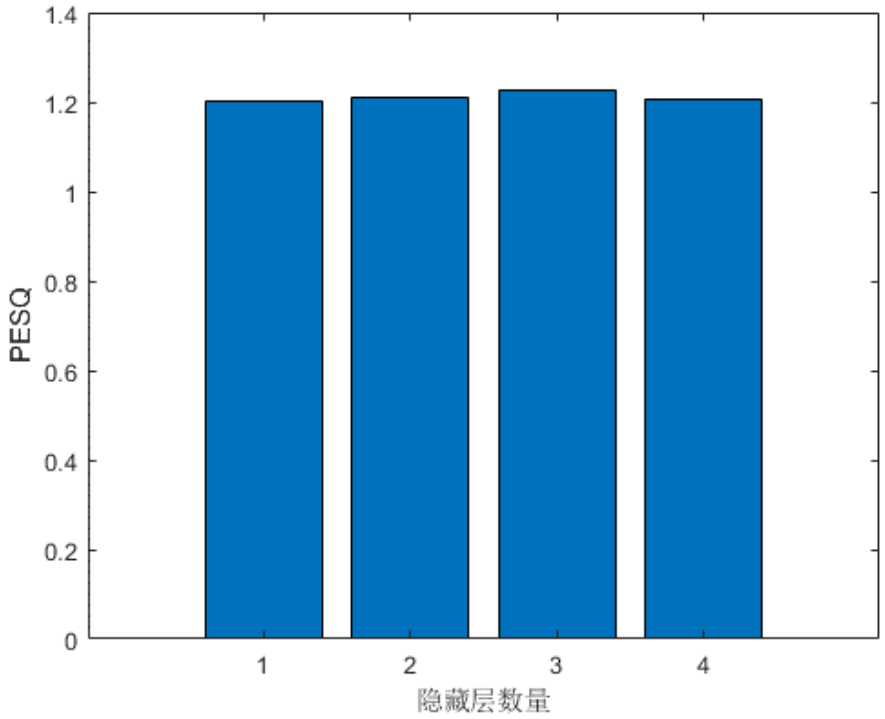

图 3-11 不同隐藏层数量下神经网络输出结果的 PESQ 值柱形图

表 3-4 不同隐藏层数量下神经网络输出结果的 PESQ 值统计表

隐藏层数量	1	2	3	4
PESQ	1.2008	1.2092	1.2250	1.2046

从数据中可以看出，三层的神经网络拥有最高的 PESQ 值，一层与两层的网络与之相比性能都有下降，猜测原因可能是神经网络复杂度不够，而当前任务映射过于复杂。而四层网络也没有三层的 PESQ 值高，因此可以得出结论，三层神经网络更加适合语音增强任务。

（四） 加入梅尔倒频谱数据后的语音增强

1. 梅尔倒频谱

要了解什么是梅尔倒频谱，首先要了解梅尔刻度的概念。梅尔刻度是一种基于人耳对等距的音高变化的感官判断而定的非线性频率刻度。具体定义上，一个常用的赫兹f转换为梅尔m的公式如下

$$m = 1125\, ln\left(1 + \frac{f}{700}\right), \qquad (3\text{-}12)$$

反变换公式为

$$f = 700\left(exp\left(\frac{m}{1125}\right) - 1\right), \tag{3-13}$$

而梅尔倒频谱则是一段语音信号在梅尔刻度下功率谱的离散余弦变换，其在赫兹刻度上是非线性的，但在梅尔刻度上则是线性的，更加接近人类非线性的听觉系统，因此广泛应用在音频压缩技术当中。最近随着机器学习的推广，梅尔倒频谱在语音识别方面也有了重要地位。

计算梅尔倒频谱的方法如下：

1) 计算信号的短时傅里叶变换，方法同式（3-3）。
2) 计算梅尔刻度下的滤波器组。本论文需要 13 个最终参数，即需要 13 个滤波器，那么需要 15 个采样点。论文中数据集采样率是 16kHz，根据奈奎斯特准则，最高可记录 8kHz 的声音频率，我们把这 20Hz-8kHz 频率根据式（3-12）转换到梅尔尺度：31.69mel-2835mel，然后将其分成 14 份，这样算上头尾就有了 15 个点。再把这 15 个点转换回赫兹尺度，记此时得到的 15 个点为$h(i)$，利用公式

$$f(i) = floor\big((x+1) \times h(i) \,/\, samplerate\big) \tag{3-14}$$

计算滤波器要用到的 15 个点。第一个滤波器从第一个点开始，在第二个点到达最大幅度，在第三个点回到 0；第二个滤波器从第二个点开始，在第三个点到达最大幅度，在第四个点回到 0，以此类推。滤波器的数学形式如下

$$H_m(k) = \begin{cases} 0, \ k < f(m-1) \ \ or \ \ k > f(m+1) \\ \frac{k - f(m-1)}{f(m) - f(m-1)}, \ f(m-1) \le k \le f(m) \\ \frac{f(m+1) - k}{f(m+1) - f(m)}, \ f(m) \le k \le f(m+1) \end{cases}. \tag{3-15}$$

其中m是第几个滤波器，最终滤波器的频响（以 10 点为例）如图 3-9 所示，图中显示了 10 个滤波器。

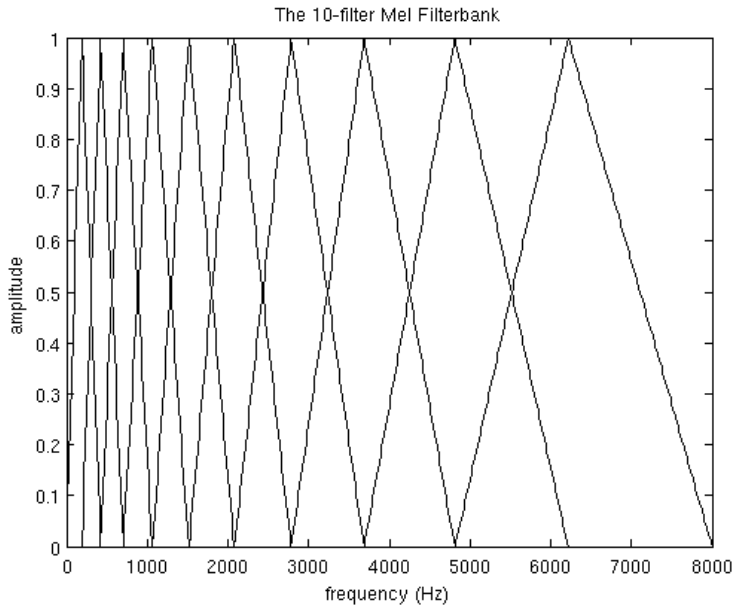

图 3-12 梅尔滤波器组的频响

然后计算 13 个滤波器内的能量，只需把每个滤波器与当前帧的频谱点乘即可，最终即可得到 13 个能量值。

3) 取这 13 个能量值的离散余弦变换，得到的结果（依然是 13 个值）即为语音数据的梅尔倒频谱。

另外，我们还需要计算它们的一阶差分与二阶差分，也将在稍后加入神经网络的训练中去。其计算方法为

$$d_t = \frac{\sum_{n=1}^{N} n(c_{t+n} - c_{t-n})}{2\sum_{n=1}^{N} n^2}. \qquad (3\text{-}16)$$

其中，左边即为第t帧的一阶差分值，右边N表示前后 N 帧，其典型值为 2，本论文也将使用这个值。而二阶差分即为一阶差分的一阶差分，计算方法同式（3-16）。以上所有计算已包含进 Python 的 python_speech_features 库内，代码中直接调用 mfcc、delta 方法即可。

2. 共享神经网络的多目标准则学习

前面我们计算了语音信号的梅尔倒频谱数据及其一阶差分和二阶差分，一共$3 \times 13 = 39$个数，现在我们将它加入到我们的神经网络的输入与输出中来，与之前的加帧组后的语谱数据同时参与网络的训练优化，之前我们只使用归一化后的对数功率谱的最小均方误差作为代价函数，而现在我们还要使用 MFCC 及其一、二阶差分值的最小均方误差作为代价函数，Adam 优化器将同时优化这两个值，想借此给原有的神经网络一些辅助作用。因为我们的神经网络是全连接的，所以帧组的输入与帧输出的部分，在训练的时候也会影响 MFCC 部分，反之亦然。图 3-10 展示了这个过程。

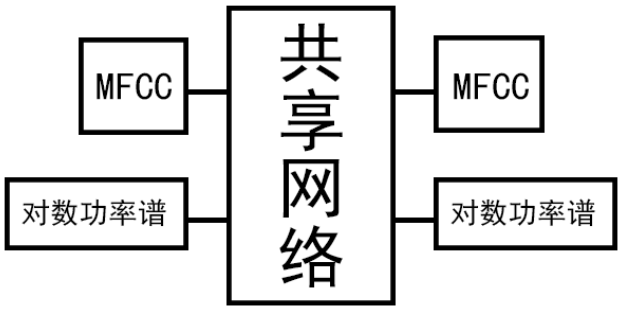

图 3-13 共享神经网络示意图

我们选取 7200 条信噪比-5dB 的语音数据作为训练集，三层隐藏层，每层神经元 2048 个，帧组内帧数为 11 帧，得到结果如图 3-11 与表 3-5。

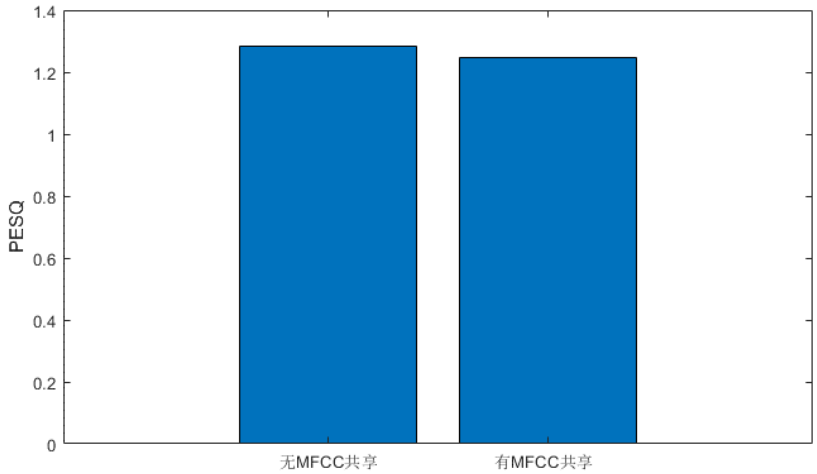

图 3-14 有无 MFCC 共享的 PESQ 值对比柱形图

表 3-5 有无 MFCC 共享的 PESQ 值对比统计表

是否有 MFCC 共享网络	否	是
PESQ	1.2826	1.2461

在有 MFCC 参与共享神经网络的情况下，数据统计结果显示，其效果竟然不如只使用对数功率谱的神经网络，这是与我们预想的完全不同的。这方面的原因有待探究。

四、 结论与展望

经过上面的定量分析，我们已经给出了神经网络的不同结构对输出的结果影响。除了帧组内帧数这个因素外，其他都得到了比较符合直觉的结论。需要注意，研究的目的不只是得出数字上的结论，更多的还要依据此来选择特定情况下更合适的神经网络。

而对于帧组内帧数的“反直觉”的结论，或许我们能从训练集数量的影响中看出一点眉目。我们注意到随着训练集数量从 900 条语音上升到 7200 条语音，网络输出结果的 PESQ 值是在上升的，所以我们有理由认为，3600 条训练集是不够的，即认为我们的神经网络处于欠拟合状态。另外，考虑到增加输入帧数会增加网络的复杂度，所以我们可以大胆猜测，具有更复杂的全连接神经网络，需要有更多的训练集数据作为支撑，才能达到更加优秀的效果。在后来的测试中，我们尝试把 7200 条语音数据送入 11 帧作为帧组输入的神经网络，结果得到了比之前 3600 条还要好的 PESQ 值（1.2826，之前 3600 条语音的结果是 1.2050，这个结果我们稍后还将用到），更是证明了我们的猜想。

而在将语音的 MFCC 数据加入到网络输入层，共同参与训练后，观察到网络输出的 PESQ 结果并不令人满意，原因尚不明。

我们还找到一些其他研究者在利用神经网络进行语音增强的尝试，其中印象比较深刻的是 Mozilla 公司发表的有关利用循环神经网络进行语音增强的论文[4]。论文中指出了循环神经网络对语音增强的重要性，因为其可以对时间序列进行建模，而不是单独考虑输入和输出帧，从而有足够的时候来对噪声产生一个良好的估计值。曾经很长一段时间，循环神经网络因为无法保存过长时间的信息，而并没有展现出很好的性能，但后来在包括长短期记忆网络（LSTM）、门控递归单元（GRU）等门控单元发明后，这个问题得到了很好的解决。而该论文使用的就是后者提到的 GRU，它比 LTSM 表现更好，且占用资源更少。如图 3-12 所示，最左边的是前馈单元，也就是我们之前用的神经网络里的神经元，中间的是最简单的递归神经单元，右边则是门控递归单元。后者的区别在于有两个门：r 和 z 门，都是软开关，机遇正隔层的前一个状态和输入计算得到，当更新门 z 在左端时，状态可以在很长的一段时间保持不变，直到某个条件使得 z 门接通右侧，使得 GRU 可以学习长期的模式。

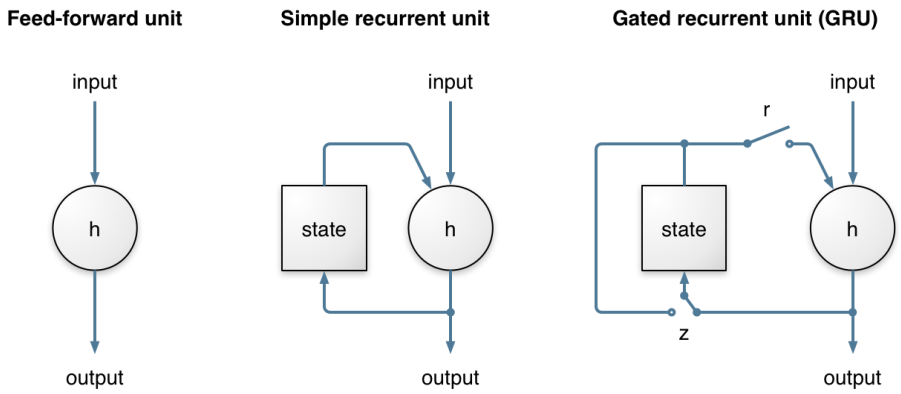

图 3-15 前馈网络、递归单元、门控递归单元

同时，在他们的论文中，他们使用了类似梅尔尺度的“Bark 尺度”，并只将 Bark 倒频谱（BFCC）及其一阶、二阶差分数据送入网络训练，从而得到带噪语音在不同频段内应匹配的增益，最后将这些增益与频谱相乘即得到增强后语音。在没有帧数据参与训练的情况下，该方法可以使用极少的硬件资源进行训练，并且在训练完成后移植到 C 语言，可以在树莓派上进行实时的语音增强。

参考文献

[1] https://en.wikipedia.org/wiki/Mel_scale 维基百科：梅尔尺度.

[2] https://en.wikipedia.org/wiki/Mel-frequency_cepstrum 维基百科：梅尔倒频谱.

[3] https://arxiv.org/pdf/1606.04080.pdf Google DeepMind, Matching Networks for One Shot Learning.

[4] https://arxiv.org/pdf/1709.08243.pdf Mozilla Corporation, A HYBRID DSP/DEEP LEARNING APPROACH TO REAL-TIME FULL-BAND SPEECH ENHANCEMENT.

[5] https://algorithmsdatascience.quora.com/ReLu-compared-against-Sigmoid-Softmax-Tanh ReLu compared against Sigmoid, Softmax, Tanh.

[6] http://www.doc88.com/p-2337088655000.html 神经网络的函数逼近理论.

[7] 徐勇．基于深层神经网络的语音增强方法研究，2015.

后记

虽然论文中加入梅尔倒频谱的部分没有得到想要的结果，但整体还算差强人意。当中数学证明的部分与整个神经网络的知识给我带来很大的提升。Pebble 老师水平也很高，每次找他聊论文内容，他都能给我很好的建议，让我醍醐灌顶。还要感谢我的朋友，澳门科技大学的 alprc，带我入坑机器学习。“子贡倦于学，告仲尼曰：‘愿有所息。’仲尼曰：‘生无所息。’子贡曰：‘然则赐息无所乎？’仲尼曰：‘有焉耳，望其圹，皋如也，宰如也，坟如也，鬲如也，则知所息矣。’子贡曰：‘大哉死乎！君子息焉，小人伏焉。’” 因此还要感谢我的某些把毕设当作负担的同学，他们的糊弄与不在意时刻都在提醒我，让我认真对待毕设，努力提高自己。

另外，本论文是第一次开有关神经网络的内容，初次尝试必然会有很多不足，我在这里也为后来者留些建议，个人认为参考文献[7]的论文不是非常有用处，而参考文献[4]的内容则非常让我震惊，其使用了循环神经网络，性能还非常强，论文中有网页版的实时处理 demo，性能非常惊人。由于发现这篇论文的时候已经离论文交稿日很近，所以没有在此篇论文上多做研究，不过在我的代码中已经留有与 BFCC 有关的内容，在此，首先对我论文的不周全表示抱歉与遗憾，其次强烈建议后来者对参考文献[4]详细研究，也希望我写的代码（虽然没用上）可以助你一臂之力。