ALGE-A10 TM1000 基于神经网络的解调精度增强技术研究

文档编号：ALGE-A10-014

版本信息：V1.0

编　　制：李日

审　　核：李玉宝

批　　准：祝希

2025-10-19　发布　　2025-10-19 实施

文档变更历史

版本	变更日期	撰写	审核	变更描述
V1.0	2025-05-26	李日		初始版本
V1.0	2025-10-19	李日		版本管理

目录

1 前言

1.1 目的

将接收符号软解调（解映射）为比特对数似然比（LLR）是现代通信接收机的核心功能。目前常见的软解调方法为 log-map 和 max-log-map，后者是对前者的一种简化运算，但牺牲了性能；本文旨在通过一种神经网络的架构去实现 log-map，预期以低计算复杂度获取接近最优性能。

1.2 范围

该研究报告涵盖 1T2R 功能，具体包括以下内容:

- 1 收 1 流场景下的接收解调处理。
- 2 收 1 流场景下的接收解调处理。
- 2 收 2 流场景下的接收解调处理。

1.3 参考文献

- Ori Shental, Jakob Hoydis, ""Machine LLRning":Learning to Softly Demodulate"

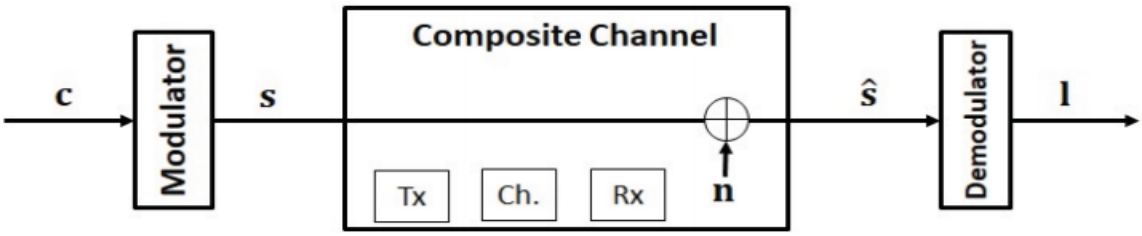
1.4 术语、定义和缩略语

缩略语	英文全称	中文全称
LLR	Log Likelihood Ratios	对数似然比
NN	Neural Network	神经网络
QAM	Quadrature Amplitude Modulation	正交幅度教条
AWGN	Additive White Gaussian Noise	加性高斯白噪声

1.5 软件版本基线

本规范文档不涉及软件代码

2 LLR 计算



如上为简单的通信系统，包括信源信息c，调制符号s，过信道（加性高斯白噪声n），接收符号$\hat{s}$，最后解调输出软信息l。

$$\hat{s} = s + n$$

其中高斯白噪声n服从分布为

$$n \sim CN(0, \sigma^2)$$

2.1 常用的解映射方法

在获取接收信号$\hat{s}$后，终端需要通过解调器将符号解映射到比特级 LLR，最后送给译码器。依据最大后验概率（Maximum a Posterior probability）准则，即希望在接收信号为$\hat{s}$的情况下，最大化发送比特对应的概率比表示为:

$$l_i = \log \frac{\mathrm{P}(c_\mathrm{i} = 0|\hat{s})}{\mathrm{P}(c_\mathrm{i} = 1|\hat{s})}, \qquad i = 1, \dots, M.$$

l_i表示第i个比特 LLR，M为调制阶数，即一个调制符号所对应的比特数。

对上式基于高斯分布概率密度函数的公式进一步展开可得 log-map 算法表达式：

$$l_i = \log \frac{\sum_{s \in C_i^0} \exp(-\frac{||\hat{s} - s||_2^2}{\sigma^2})}{\sum_{s \in C_i^1} \exp(-\frac{||\hat{s} - s||_2^2}{\sigma^2})}, \qquad i = 1, \dots, M.$$

式中C_i^0表示当前调制阶数下，第i个比特为 0 对应的所有调制符号的集合。

已知数学中常用的 max 函数为：

$$\log\left(\sum\nolimits_j \exp(-x_i^2)\right) \approx \max_j(-x_i^2)$$

则代入 log-map 公式可得 max-log-map 算法公式：

$$l_i = \frac{1}{\sigma^2}\left(\min_{s\in C_i^1}\left(||\hat{s}-s||_2^2\right) - \min_{s\in C_i^0}\left(||\hat{s}-s||_2^2\right)\right)$$

可以观察到该算法可以避免复杂的指数与对数计算，实现简单，但性能会有损失。

2.2 基于神经网络的方法

2.2.1 神经网络结构

以下为神经网络的方法来拟合 log-map 函数处理，以期获取相近的性能。

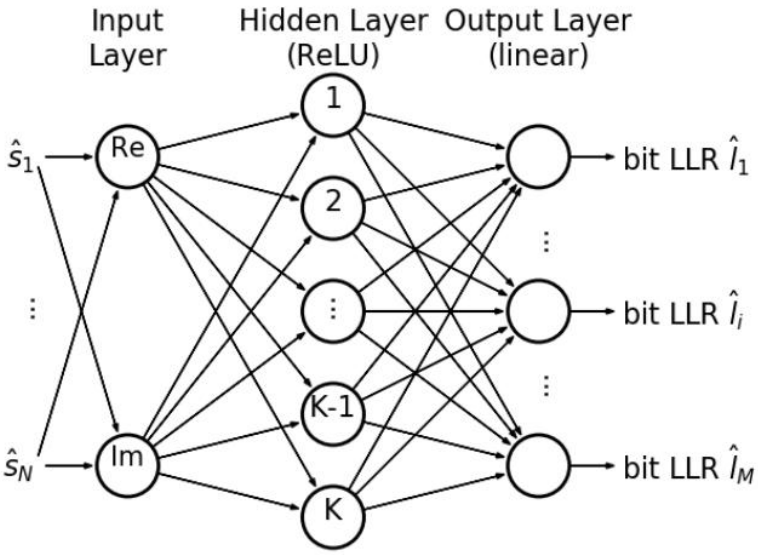


图 2-1 LLRnet 网络结构

可以观察到上图 LLRnet 为一简单的 NN 网络，包含输入层、隐藏层以及输出层：

- 输入层为接收到的调制符号$\hat{s}$，共两个节点，表示实部和虚部；
- 隐藏层共有 K 个节点，点数越多表示调制阶数越大；
- 输出层为 LLR，共包含 M 个节点，表示 M 个比特 LLR。

2.2.2 数据处理

下图所示为一典型 LLRnet 的网络结构拓扑图，调制方式为 64QAM，隐藏层有 16 个神经单元。

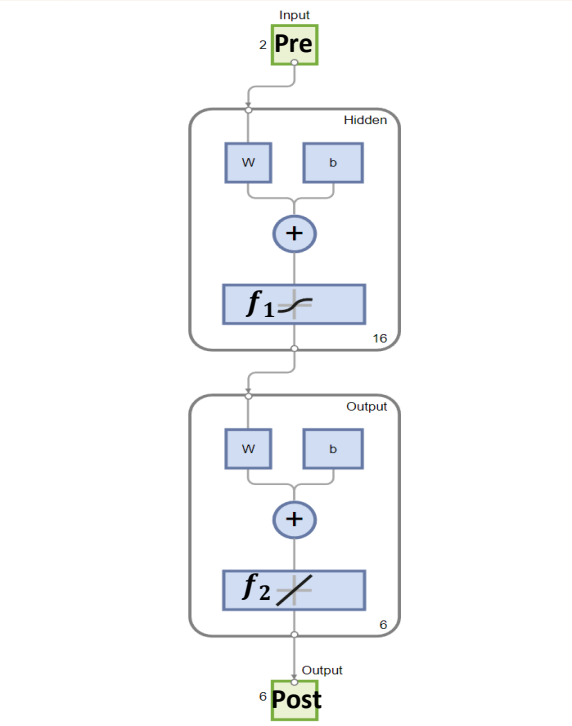


图 2-2 64QAM LLRnet 数据流程图

- 神经单元处理

  对于中间隐藏层的 K 个神经单元

$$y_k = \mathrm{f}\left(w_k^T x_k + b_k\right)$$

  式中w_k^T, b_k表示输入数据节点到隐藏层的第 k 个神经单元对应的权重和偏置，f表示激活函数。从隐藏层到输出层的数据处理同上。

- 预处理和后处理

  Mapminmax 函数会默认将数据归一化到[-1,1]，以提高神经网络的训练效率和性能

  预处理：

$$Y_{pre} = \frac{(y_{max} - y_{min})(x - x_{min})}{(x_{max} - x_{min})} + y_{min}$$

  式中

$$y_{max} = 1; y_{min} = -1$$
$$x_{max} = \max(input_X);\ x_{min} = \min(input_X);$$

  后处理：

$$Y_{post} = \frac{(x_{max} - x_{min})(y - y_{min})}{(y_{max} - y_{min})} + x_{min}$$

  式中

$$y_{max} = 1; y_{min} = -1$$
$$x_{max} = \max(output_X);\ x_{min} = \min(output_X);$$

  值得注意的是，该函数不支持非有限数值，如(NaN、Inf)，所以使用时需要对数据集进行数据的清洗。

- 激活函数

  Tanh

$$f_1(x) = \tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$$

Linear

$$f_2(x) = x$$

值得注意的是，由于最终输出 LLR 值可正可负，故激活函数需考虑数据特征。

2.2.3 网络训练

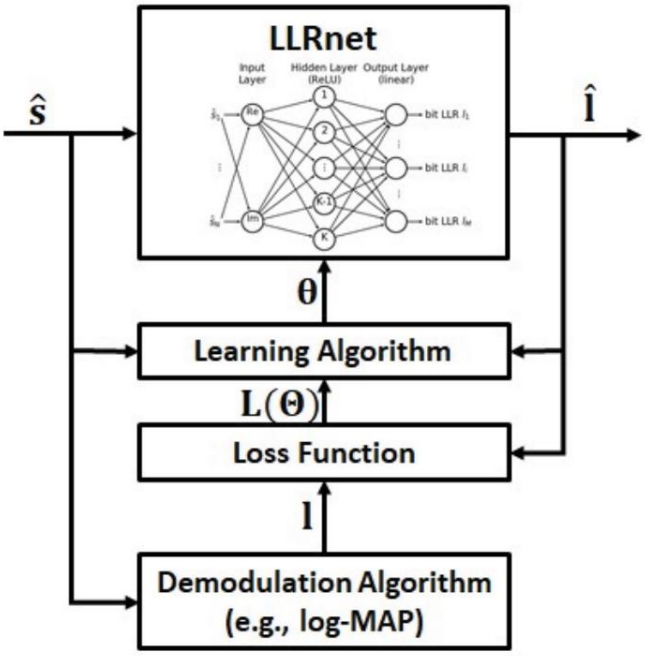


图 2-3 LLRnet 训练流程图

- 数据集

1. 数据生成

训练采用监督学习的方法，由于网络的输入数据为经过高斯白噪声信道的接收信号，所以数据集的产生为特定噪声、特定调制方式下的接收数据。

场景	取值
调制方式（2 种）	64QAM，256QAM
信噪比（36 种）	0~35dB

考虑到在 QPSK 和 16QAM 场景下，log-map 与 max-log-map 性能相差较小，故 LLRnet 重点针对高阶调制下的性能调优，这样待生成的数据集为 72 组。

输出数据集为 log-map 算法得出的 LLR 数据。

2. 数据清洗

在实际采用 log-map 计算 LLR 时，尤其是高 SNR 下，会出现 LLR 为特别大甚至无穷大的情况，对于神经网络而言，这种数据学习难度过大，影响学习效率；所以会考虑设置经验值，确保不影响解调性能的前提下降低拟合的复杂度。

3. 数据划分

70%数据用于训练，15%用于验证，15%用于测试。

- 损失函数

1. 均方误差（MSE）

$$L^{(MSE)}(\theta) = \frac{1}{B}\sum_{b=1}^{B} ||\hat{I}^{(b)} - I^{(b)}||_2^2$$

2. 交叉熵函数

$$L^{(CE)}(\theta) = -\frac{1}{B}\sum_{b=1}^{B}\sum_{m=1}^{M}(l_m^{(b)} \log\left(\hat{l}_m^{(b)}\right) + (1 - l_m^{(b)}) \log\left(1 - \hat{l}_m^{(b)}\right))$$

➢ 训练算法

1. 梯度下降法

$$\theta \leftarrow \theta - \alpha\nabla_\theta L(\theta)$$

上式中α为学习率，满足$\alpha > 0$；$L(\theta)$为损失函数；θ为参数集，即用得到的误差对参数集求偏导，并采用一定的学习率使得参数集向误差下降最快的方向变化。

2. 高斯-牛顿法

$$\theta \leftarrow \theta - \left(J^T(\theta)J(\theta)\right)^{-1} * J^T(\theta)\gamma$$

上式中J是残差γ关于θ的雅可比矩阵；

通过求解线性方程组：

$$\left(J^T(\theta)J(\theta)\right)\Delta\theta = -J^T(\theta)\gamma$$

获取$\Delta\theta$，从而进一步更新θ。

3. LM 算法

$$\theta \leftarrow \theta - (J^T(\theta)J(\theta) + \lambda I)^{-1} * J^T(\theta)\gamma$$

上式中$J(\theta)$代表目标函数关于θ的雅可比矩阵（Jacobian matrix）；γ代表残差；I 是单位矩阵，确保矩阵可逆非病态；λ是阻尼因子来控制更新步长：当 λ 较大时，算法行为类似梯度下降（稳定但收敛慢），当 λ 较小时，算法行为类似高斯-牛顿（收敛快但对初始值敏感）。

学习停止准则：

1. 所有的数据已完成训练；
2. 达到预期的误差值。

2.2.4 链路仿真

➢ 配置参数

表 2-1 仿真参数配置

配置参数	取值
带宽	10M
RB 数	51
SCS	15k
收天线数	2

流数	2
信道编码	LDPC
MCS table	Max-64QAM; Max-256QAM
MCS(low/high)	Low 为低 MCS; High 为高 MCS
信道模型	TDL-A
信道估计算法	LMMSE
MIMO 检测算法	LMMSE
译码算法	BP
AI	基于 AI 的 LLR 计算方法
Ref	基于 max-log-map 的 LLR 计算方法

➢ 仿真结果

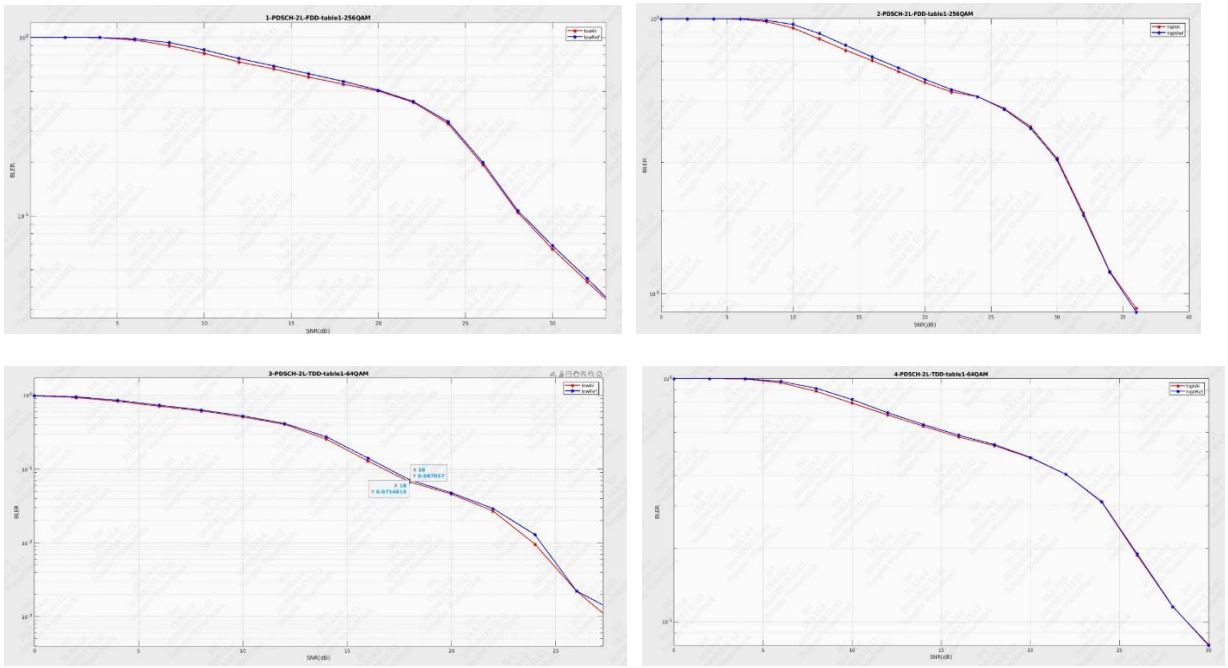


图 2-4 AI vs Rule 仿真结果

➢ 仿真观察

- 在 64QAM 低 MCS 场景下，工作点（10% BLER），AI 相比 Ref 有 5%以上的增益；
- 在高 MCS、高 SNR 场景下，性能增益不明显，有的 SNR 点会有损失；
- 在低 SNR 场景下，AI 增益较为明显，因为低 SNR 下 LLR 数据幅值较小，学习难度低。

2.2.5 遗留问题

- 拟合性增强
  - 在性能不好的场景下对网络模型做进一步的微调；
  - 增加神经网络输入层的特征，如幅值、均值等；
  - 考虑优化激活函数，增加非线性拟合的能力；
  - 增加隐藏层神经单元数或隐藏层数，增强网络的解释能力。
- 泛化性增强
  - 考虑将 SNR 信息喂给神经网络，适当降低模型数量；
  - 增加数据集，考虑从仿真平台收集 LLR 数据信息；
  - 减少过拟合，考虑通过 drop out、正则化等方法。

3 总结及结论

本文介绍了一种基于神经网络计算 LLR 的方法（LLRnet），即采用神经网络来拟合 log-map 算法，从而达到优化链路性能的目的:

- 经仿真验证，在高阶调制、低 MCS、低 SNR 的场景下，该网络相比于 max-log-map 有一定的性能增益，但仍需优化；
- 网络中所涉及数据处理均为简单的乘、加运算，激活函数的非线性也可以考虑分段拟合的方式予以简化；
- 网络可以采用线下训练的方法，且无需依赖复杂的通信链路就可以生成数据集，增加泛化性。

最后文章也提出了一些后续遗留问题，可以从拟合性角度提高网络的解释能力，提升性能；同时从泛化性角度降低网络的复杂度，便于实现。