实验一：基于广义线性模型的命名实体识别实现

邓雅萱　学号：1120202004　教学班号：07112004

一、实验目的：命名实体识别

1.1 实验要求

命名实体识别，即对文本中具有特定意义的词进行识别，在自然语言处理领域具有基础性意义。本实验通过构建广义线性模型，实现了对人名、地名、机构名三类实体的识别，并采用 BIO 表示法进行了相应的标注。

在 BIO 表示法中，B 代表 Begin，即实体的开始；I 代表 Inside，即实体的内部；O 代表 Outside，即实体的外部。以李华喜欢在北理工的图书馆自习为例，其 BIO 标注如图1-1所示。

图 1-1 BIO 表示法示例

现给定北京大学《人民日报》基本标注语料库，本实验要求建立模型解决以下问题：

(1) 将语料库数据表示为数学向量，生成训练集、验证集、测试集。
(2) 构建广义线性模型，通过梯度下降法求解相关参数。
(3) 根据验证集的结果对模型进行优化，通过测试集预测的结果来评估模型的准确性。

1.2 实验方法

本实验主要通过如下步骤，实现了命名实体识别的目标：

(1) 对原始数据集进行向量化表示，选取特征将每个词语表示为一个向量，并将对应的分类标签转化为可以参与运算的数字。
(2) 生成训练集、验证集和测试集，并对训练集中的数据进行数据清洗。
(3) 初始化参数 W，确定学习率和迭代次数，重复步骤 (4) 到 (7) 直至达到选定的迭代次数：
(4) 采用 mini-batch 方法，抽取一定数量的训练集数据，计算 Softmax 概率。对于一个输入，概率最大的即为当前模型所预测的分类。

(5) 计算当前模型的损失函数值，并对其求导。

(6) 根据损失函数的导数，采用梯度下降法对参数 W 进行更新。

(7) 将模型预测的分类与实际分类进行比较，求出用于评估当前模型精确程度的训练集与验证集的 $F1-measure$，并保存使得验证集 $F1-measure$ 最大值及对应的参数 W。

(8) 通过改变输入词的特征选取、训练集中数据清洗的方式、训练时的学习率和迭代次数、参数 W 更新的方法，比较其对应的验证集 $F1-measure$ 的大小，选择最好的模型用于预测。

(9) 将对应模型在步骤 (7) 保留的 W 值用于测试集，计算其 $F1-measure$。

全过程流程图如图1-2所示：

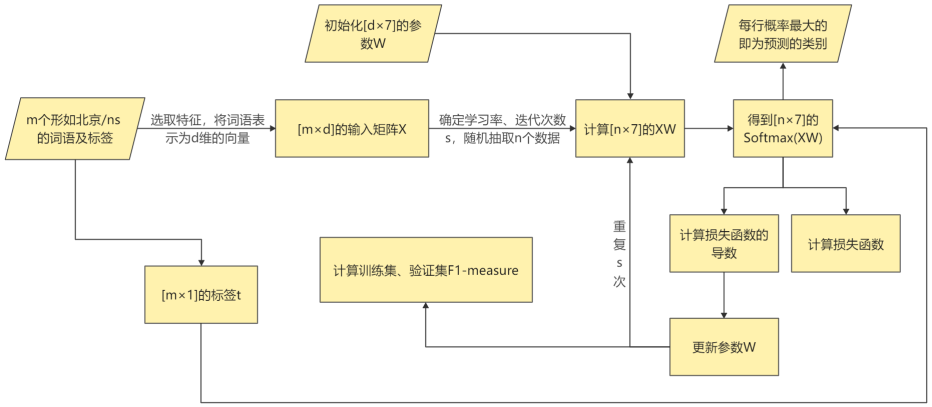


图 1-2 命名实体识别全过程流程图

目录

二、实验原理：广义线性模型

针对七分类问题，采用广义线性模型进行预测。

首先，引入指数分布族，其形式如下：

$$\mathrm{P(y};\eta) = \mathrm{b(y)}\exp\left(\eta^T T(y) - a(\eta)\right) \tag{2.1}$$

其中，η 为分布的自然参数，$T(y)$ 是充分统计量。当参数 a、b、T 均固定时，$P(y;\eta)$ 是以 η 为参数的分布族。

通过下述三个假设，指数分布族可以被转换为广义线性模型。

(1) $\mathrm{y} \mid \mathrm{x};\theta \sim \mathrm{ExpFamily}(\eta)$：给定样本 x 和参数 θ，分类 y 服从指数分布族中的某种分布。

(2) $h_\theta(x) = E[T(y) \mid x;\theta]$：给定 x，目标函数 $h_\theta(x)$ 为 $T(y)$ 的期望值。

(3) $\eta = \theta^T x$：人为构造，意在用成线性关系的来表示 η。

接下来，首先验证本实验分类的分布属于指数分布族，并根据上述假设构造广义线性模型。假设其数据分布符合多项分布，即 $y \mid x;\theta \sim \mathrm{Multi}\left(\phi_1, \phi_2, \ldots, \phi_7\right)$，$\sum_{i=1}^{7}\phi_i = 1$。

y 的概率分布如式2所示：

$$p(y \mid x;\theta) = \prod_{i=1}^{7}\phi_i^{I_{y=i}}$$

通过化简，得到

$$p(y \mid x;\theta) = \exp\left(\begin{pmatrix}\log\frac{\phi_1}{1-\sum_{i=1}^{6}\phi_i}, \ldots, \log\frac{\phi_6}{1-\sum_{i=1}^{6}\phi_i}\end{pmatrix}\begin{pmatrix}I_{y=1}\\ I_{y=2}\\ \cdots\\ I_{y=6}\end{pmatrix} + \log\left(1-\sum_{i=1}^{6}\phi_i\right)\right)$$

其中，

$$\begin{cases} b(y) = 1 \\ \eta = \left(\log\frac{\phi_1}{1-\sum_{i=1}^{6}\phi_i}, \log\frac{\phi_2}{1-\sum_{i=1}^{6}\phi_i}, \ldots, \log\frac{\phi_6}{1-\sum_{i=1}^{6}\phi_i}\right) \\ T(y) = \left(I_{y=1}, I_{y=2}, \cdots, I_{y=6}\right)^T \\ a(\eta) = -\log\left(1-\sum_{i=1}^{6}\phi_i\right) \end{cases}$$

符合式 (2.1) 的形式，故多项分布属于指数分布族。

令 $\eta = \theta^T x$，可以求得参数 ϕ 与 x 的关系：

$$\phi_j = \frac{e^{\theta_j^T x}}{\sum_{i=1}^{k} e^{\theta_i^T x}}, \quad 1 \le j \le 7$$

根据假设 (2)，可得目标函数

$$\begin{aligned} h_\theta(x) &= \mathrm{E}[T(y) \mid x;\theta] \\ &= \prod_{j=1}^{7} \phi_j^{I_{y=j}} \\ &= \left(\frac{e^{\theta_j^T x}}{\sum_{i=1}^{7} e^{\theta_j^T x}} \right)^{I_{y=j}} \end{aligned} \tag{2.2}$$

为了使得优化目标最大，可采用最大似然估计的方法求解参数 θ，该过程等价于最小化似然函数的相反数。据此，可得 Softmax 损失函数（又称交叉熵误差）。

$$l(\theta) = -\frac{1}{m} \sum_{l=1}^{m} \ln \left(\prod_{j=1}^{7} \left(\frac{e^{\theta_j^T x}}{\sum_{i=1}^{7} e^{\theta_i^T x}} \right)^{I_{y=j}} \right) \tag{2.3}$$

其中，m 为样本数量。

对损失函数进行求导，由链式法则

$$\frac{\partial l(\theta)}{\partial \theta} = \frac{\partial l}{\partial \phi} \cdot \frac{\partial \phi}{\partial \theta} = -\frac{1}{m} \sum_{l=1}^{m} [\phi_1, \cdots, \phi_j - 1, \cdots, \phi_7]^T \cdot x \tag{2.4}$$

对于每一个特征所对应的参数参数 θ_j，通过梯度下降法更新参数

$$\theta_j := \theta_j - \alpha \frac{\partial l(\theta)}{\partial \theta} \tag{2.5}$$

根据指数分布族的性质，可以知道该模型的极大似然对数是凸函数。因此，利用上述方法，多次迭代即可得到全局最优解。

三、实验内容：Softmax 回归的建立

对实体进行命名实体识别，实质上是对每个词进行七分类，即 B-人名、I-人名、B-地名、I-地名、B-机构名、I-机构名、O 七类。针对多分类问题，可采用广义线性模型中的 Softmax 回归进行求解。

3.1 数据预处理：one-hot 表示

在 Softmax 回归中，输入为向量，输出为对应的分类。由于在本实验中，数据集的形式为中文文本及每个词语对应的标签，因此首先需要对数据进行预处理。

对于**中文文本**，本实验中采用 one-hot 表示法，每一个词语对应于一个 1×1000 维的向量：[词 1，$\cdots$，词 999，未出现]。其中，为了更好地对文本特征进行描述，以 **1:1:1:1** 的比例选取训练集中出现频率最高的人名、地名、机构名与 O 类词共同构成该向量前

999 维（O 类词选取 249 个）。同时，为了提升文本特征质量，使用了中文停用词表等四份**停用词库**对标点、阿拉伯数字等词语进行过滤，最终得到作为特征表示的 999 词，词云如图3-1所示：

图 3-1 特征选取词云

在该模型学习与推理的过程中，每一个输入由当前词语及其前后词语拼接而成的 1×3000 维向量构成。针对句首和句尾词的空缺处，采用 1×1000 维的 0 向量进行填充。

对于每个输入所对应的**标签**，根据数据集所附的《北大语料库加工规范：切分 · 词性标注 · 注音》，可以将其转化为实验目标的七种标签，并采用阿拉伯数字进行表示，对应关系为：0 表示 B-人名、1 表示 I-人名、2 表示 B-地名、3 表示 I-地名、4 表示 B-机构名、5 表示 I-机构名、6 表示 O 类词。针对方括号 [ ] 表示的嵌套类，仅考虑方括号外类别，并对少量不完整的方括号噪声进行删除处理；针对由大括号 { } 表示的注音标注，为还原真实文本，对其进行删除处理。

经上述处理后，得到训练集 732289 条，验证集 257810 条，测试集数据 148152 条。

在实际学习过程中，由于文本中绝大部分词语的标签均为 O，模型不仅学习速度缓慢，而且对六种分类的学习比例相对较少，导致用于评估模型好坏的 $F1-measure$ 较低。因此，需要对训练集中的 O 类词语进行**数据清洗**，本实验保留出现在 999 词集的 O 及前后非 O 类的 O 词作为输入，并保留除此以外的 30000 个 O 类词，将其余词语进行剔除处理，得到训练集数据 243607 条。

该部分的源代码见 data preprocessing.py 文件。

3.2 模型实现

设每个分类的 θ 按行排列，构成 3000×7 维的矩阵 W。设每个输入 x 按行排列，构成 $m \times 3000$ 维的矩阵 X。

3.2.1 Softmax 输出

首先，以期望为 0，方差为 0.01 的随机值对 W 进行初始化。通过 XW 的矩阵乘法计算 $\theta^T x$ 的值，并将结果代入式 (2.2)，求得 Softmax 的输出值。计算输出值时，由于需要进行指数运算，可能存在溢出问题。通过分析该函数的性质，可知

$$\phi_j = \frac{e^{\theta_j^T x}}{\sum_{i=1}^{k} e^{\theta_i^T x}} = \frac{Ce^{\theta_j^T x}}{C\sum_{i=1}^{k} e^{\theta_i^T x}} = \frac{e^{\theta_j^T x + lnC}}{\sum_{i=1}^{k} e^{\theta_i^T x + lnC}} = \frac{e^{\theta_j^T x + C'}}{\sum_{i=1}^{k} e^{\theta_i^T x + C'}}$$

在进行运算时，加上常数 C' 对结果不会产生影响。因此，通过**减去**每一个输入中的最大值，可以更安全地完成 Softmax 的运算。根据输出的结果，利用式 (2.3) 来计算损失函数。为了预防模型在正确标签处预测的概率为 0 导致取 log 后为负无穷大，实际运算时增加了一个**微小值**e^{-7}，通过计算 $log(\phi_j + e^{-7})$ 的大小来保护运算。图3-2（来自 CS231n）可以较为形象地描述上述过程，本实验中，偏置设置为 0。

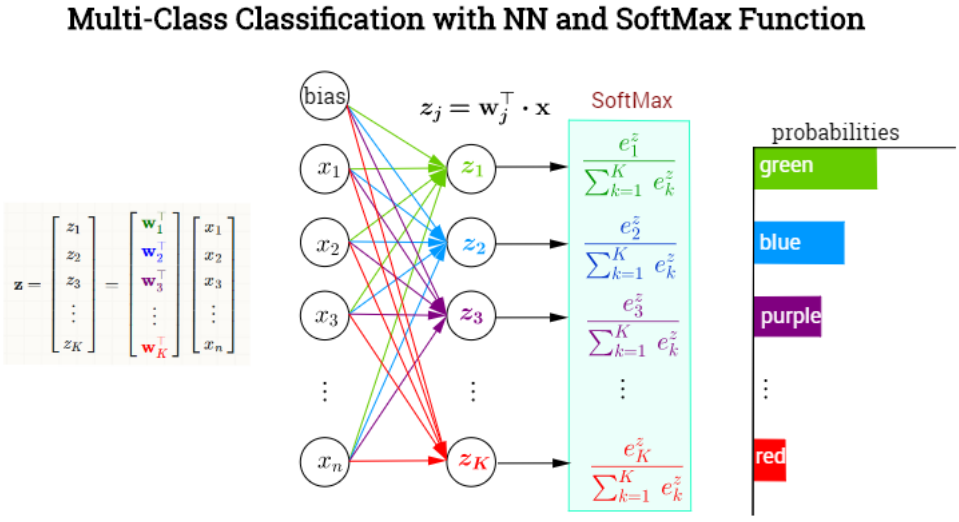


图 3-2 Softmax 过程图解

本节代码见 one_layer_net.py 中的 SoftmaxWithLoss 类中的 forward 函数。

3.2.2 梯度更新

根据式 (2.4) 计算结果对 W 的导数，具体实现见 SoftmaxWithLoss 类中的 backward 函数（手动求导）。自动求导只需将 numpy 转为 tensor，初始化 W 时使得“allow_pickle=True”，

Softmax 输出求得损失函数后再调用 loss.backward()，此时 W.grad 即为所求导数值。根据所求导数结果，由式 (2.5)，采用梯度下降法对 W 的值进行更新。由于本实验训练集样本较大，在实际运算时效率较低。综合考虑速度和准确度，选取 $\alpha = 2$ 作为学习率，并在学习过程中采取了 mini-batch 的方法。本实验设置的 batch-size 为 100，每次迭代从训练集随机抽取 100 个数据代替整个数据集作为输入，并进行 40000 次迭代。完整训练过程见 train.py 文件。

3.2.3 准确程度评价

实验中，每迭代 100 次，用所得参数对验证集中的数据进行预测，以评估模型的准确程度。由于数据集中 O 类标签比例较大，而实际需要提取的人名、地名、机构名比例较小，通过传统的 正确标签数/所有标签数 的准确率计算方法会导致准确率虚高。因此，本实验采用 $F1 - measure$ 来计算准确率。首先，计算**查准率**，即模型做出的所有实体预测中正确的比例，选取数据 Softmax 输出值最大的作为数据的预测结果，并提取预测的所有实体，与实际标签进行比较。接着，计算**查全率**，从模型的标签中提取出所有实际的实体，与模型预测结果进行比较。结合上述标准，使得模型既能找出尽可能多的实体，并做出尽可能多的正确预测，采用如下公式来评估模型预测的准确程度：

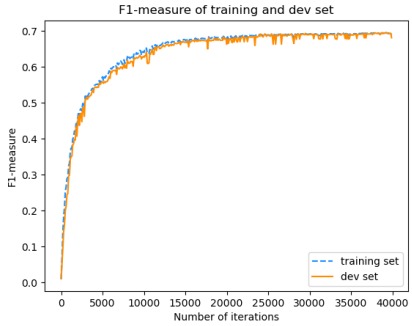
$$\text{F1-measure} = \frac{2 \cdot \text{查全率} \cdot \text{查准率}}{\text{查全率} + \text{查准率}}$$

最后，选取验证集中 $F1 - measure$ 最高的一组参数 W 用于测试集的计算，用于评估模型的准确率。对于该部分的计算，通过 one_layer_net.py 中的 SoftmaxWithLoss 类中的 accuracy 函数实现。

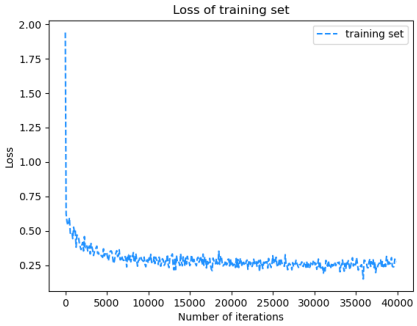
3.3 实验结果：loss 与 F1-measure 呈现

在学习率 $\alpha = 2$，迭代次数为 40000 次的条件下，验证集 $F1 - measure$ 最大时测试集结果为 67.37%，准确率较高。训练过程中的 loss 变化以及 $F1 - measure$ 变化如图3-3所示：

分析图中数据可以发现，训练集与验证集 F1-measure 较为接近，没有发生过拟合现象。同时，loss 下降较快，说明学习率的选取较为合适。但是随着迭代次数的增加，损失函数波动较大，可以采用学习率递减的 AdaGrad 算法或者更为优异的 Adam 优化器来解决上述问题。



(a) F1-measure 变化



(b) 损失函数变化

图 3-3 结果呈现

四、实验分析：参数的选取与调整

4.1 输入词的特征选取

控制学习率为 2、迭代次数为 10000 与训练集生成方式不变，对特征类别、特征数量进行改变。训练集生成方式：保留出现在特征词集的 O 及前后非 O 类的 O 词作为输入，其余 O 类词仅保留 60000 个。

4.1.1 特征类别

在该小节中，每个词所对应的向量的维度为 $[1 \times 800]$，即每个输入所对应的向量维度为 $[1 \times 2400]$。首先，选取全部文本词频最高的 799 个词语作为特征描述，接着，引入中文停用词表、哈工大停用词表、百度停用词表、四川大学机器智能实验室停用词表合并去重后形成的词表作为停用词库，并手动将部分特殊符号进行剔除处理，得到全文词频最高的 800 词，生成训练集 323850 条，其训练集与测试集的结果如下：

图 4-1 选取全文词频最高的 799 词作为特征描述时的情况

在直接选取全文词频最高的词用于表示输入的情况下，出现了一定程度的过拟合现象，这是因为所要预测的人名、地名等分类的常用词语在全文中所占比例较小所导致的，需要对特征表示词进行一定筛选。人工以近似 1:1:1:1 的比例选取人名 200 词、地名 200 词、机构 200 词、其他类 199 词**共同构成特征**，进行学习与预测，得到验证集 **67.8%** 的准确率，结果相对较前者产生大幅度改善。

4.1.2 特征数量

改变输入词向量表示的维度，分别取 500、800、1000、1200、1500，其结果如图4-2所示。

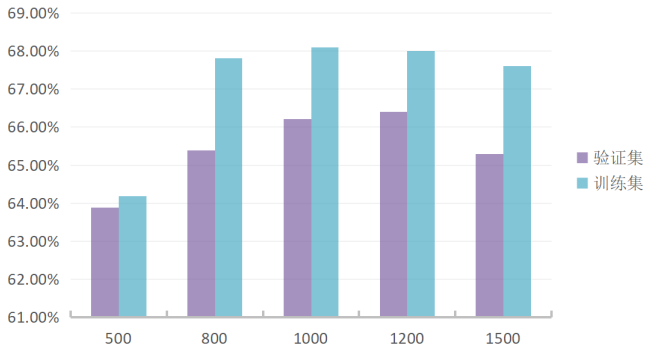


图 4-2 向量不同维度下的 F1-measure 结果

分析结果发现：验证集的向量维度由 500 增加至 1000 时，$F1-measure$ 由 63.9% 增加至 66.2%；从 1000 维增大至 1200 维时，准确率变化不大；而当向量增大至 1500 维时，$F1-measure$ 反而下降至 65.3%。从中可以看出，增大向量维度可以提高模型对词语的表现能力。然而，更大的向量维度意味着需要运用更多数据对模型的进行训练，因此，过多地增大向量维度会导致模型准确性降低。因此，在该模型中，选择 **1000 维**作为每个词所对应的向量的维度。

4.2 训练集数据清洗

由于训练集中 O 类占比较大，导致对六种分类的学习比例相对较少，不仅会使模型学习速度缓慢，而且用于评估模型好坏的 $F1-measure$ 大大降低。因此，需要对训练集中的 O 类词语进行数据清洗。本实验所采用的数据清洗方式为：保留出现在特征词集的 O 及前后非 O 类的 O 词，并保留其余一定数量的 O 类词。为了进行对比，控制学习率为 2，迭代次数为 10000，分别将保留的 O 类词控制为 10000、30000、40000、60000、90000、120000。

当增加 O 类数量在一万到三万区间，整体 $F1-measure$ 变化不大。大于 30000 时，

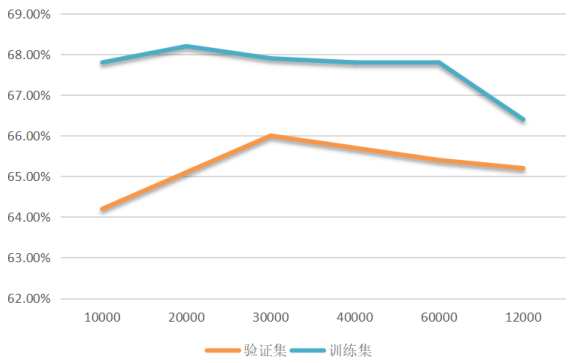
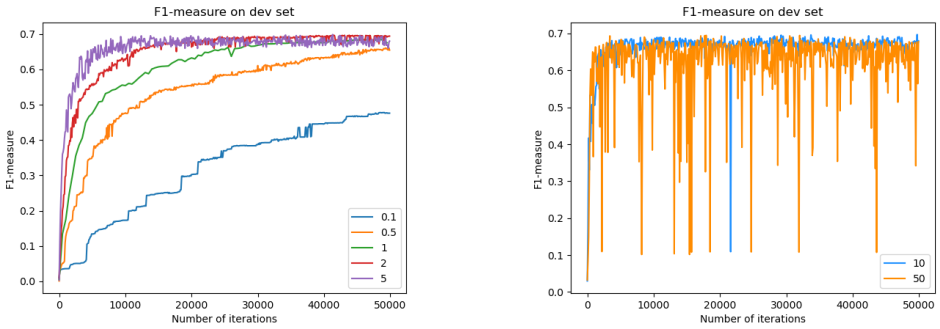


图 4-3 保留 O 不同数量下的 F1-measure 结果

整体呈下降趋势，但下降幅度相对较小，这是因为此时增加的 O 类的向量表示均相同。综合多种因素考虑，选取 **30000** 为其余保留的 O 类词个数。

4.3 学习率的选取

选取学习率为 0.1、0.5、1、2、5、10、50，在训练集为 40 万时，迭代 50000 次，结果见图4-4。从图4-4a可以发现：在学习率小于 1 时，$F1-measure$ 增加的速度较为缓慢；而当学习率逐渐增大时，每 100 次迭代时的 $F1-measure$ 变化幅度较大。从图4-4b可以看出：当学习率为 10 时，出现了迭代前后出现了 $F1-measure$ 变化超过 0.5 的情况；当学习率为 50 时，波动更大。



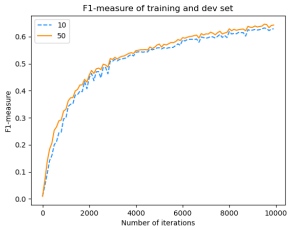
(a) 学习率为 0.1、0.5、1、2、5　　**(b) 学习率为 10、50**

图 4-4 不同学习率下的 F1-measure 结果

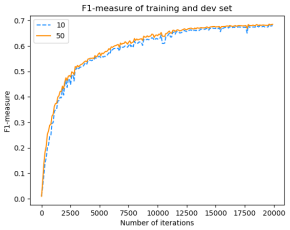
由于本实验采用了 mini-batch 的小批量下降，学习过程本身具有随机性。同时，记录验证集数据时只记录了每 100 次迭代时的情况，实际情况较图中波动更大，会导致模型无法收敛至最优值。因此，为了增加模型的鲁棒性的同时保证一定的学习效率，选取 **2** 作为模型的学习率。

4.4 迭代次数的选取

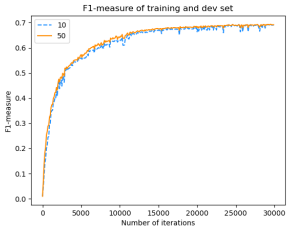
选取迭代次数为 10000、20000、30000、40000、50000，得到训练集和验证集 $F1-measure$ 的变化过程及结果如下：



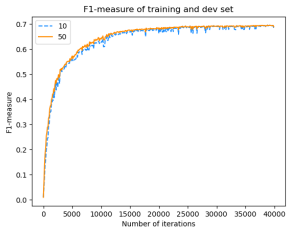
(a) 迭代 10000 次



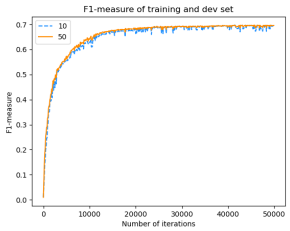
(b) 迭代 20000 次



(c) 迭代 30000 次



(d) 迭代 40000 次



(e) 迭代 50000 次

图 4-5 不同迭代次数下训练集、验证集的 F1-measure 变化过程

表 4-1 不同迭代次数下 F1-measure 的最大值

迭代次数	10000	20000	30000	40000	50000
验证集最大	9500	19600	29500	38800	49200
训练集	0.63238808	0.683385367	0.693965988	0.696113588	0.695935769
验证集	0.646582435	0.685843785	0.693533861	0.696053213	0.697104661

从图4-5和表4-1中可以发现，随着迭代次数由 10000 增加到 40000 次，验证集的 $F1-measure$ 由 64.6% 提升至 69.6%。当迭代次数大于 30000 次后，$F1-measure$ 增速明显放缓。特别是由 40000 次增加至 50000 次时，增加 10000 次迭代仅使 $F1-measure$ 增加 0.1%，变化不明显。因此，为了提高效率同时保证一定准确率，选取 **40000** 作为迭代次数。

4.5 可能的改进方案

- 本实验仅采用 SGD 算法，有时候可能效率不高或者不够精确。可以修改优化算法，尝试 RMSProp、Adam 等不同优化器。

- 本实验仅采用单层网络，未设置隐藏层。可以考虑增加神经网络层数，引入激活函数，以更好地对输入词进行特征表示。

五、实验心得：收获与不足

通过本次实验，我获得了如下收获：

(1) 对广义线性模型的理解更加深刻，从纸上谈兵到躬身实践，对损失函数、梯度下降等概念与机器学习模型建立全过程的理解更加清晰。
(2) 熟悉了 python 语言的语法，逐渐提升自身编程能力，并学会了高效利用网上资源自学。
(3) 增强了对问题全面考虑与解决实际问题的能力。由于过程中容易出错，编程时会预先考虑与发现错误后的调试；为了对数据进行分析和对学习进度进行监测，实现了到在终端显示清晰、文件方便读取的目标，完成了从做算法题时根据题目要求输出到根据自己实际需要决定输出的转变。

但由于是第一次对相关领域进行全过程的编程实践，过程中仍有不足：

(1) 由于对 python 语言的不熟悉与自身不良编程习惯，开始写代码的时候并未运用函数与类，导致后续修改代码时带来理解上的困难，最后无奈将数据预处理部分重写了一遍，会在以后编程中多加注意。
(2) 较早地输出结果却一直在盲目地调整参数，实验报告的撰写开始较晚导致过程中做了很多无用功。在下次实验中，希望能够尽早开始本报告的撰写，并在不确定的部分学着多参考网上的资料。
(3) 开始由于对模型的理解并不全面，在数据预处理时的变量命名混乱，希望之后能够明确编程的步骤以及所需要的变量。