arXiv:2305.01384v1 [cs.CL] 2 May 2023

基于类别的影响函数用于错误检测

Nguyen Duc-Thang*† Hoang Thanh-Tung*† Quan Tran*‡
Huu-Tien Dang† Nguyen Ngoc-Hieu† Anh Dau† Nghi Bui†
† FPT 软件 AI 中心 ‡ Adobe 研究院 {
nguyenducthang8a2, htt210, quanthdhcn}@gmail.com

摘要

影响函数（影响函数）是检测大规模数据集中的异常样本的强大工具。然而，将其应用于深度网络时，它们往往不稳定。在本文中，我们解释了影响函数不稳定的原因，并针对这一问题提出了一种解决方案。我们表明，当两个数据点属于不同类别时，影响函数并不可靠。我们的方法利用类别信息来提高影响函数的稳定性。大量实验表明，我们的改进方法在不增加额外计算成本的情况下，显著提升了影响函数的性能和稳定性。

1 引言

深度学习模型需要大量数据。大型模型，如 Transformer（Vaswani等人，2017年）、BERT（Devlin等人，2019年）和 GPT-3（Brown等人，2020年）需要数百万至数十亿个训练数据点。然而，数据标注是一项成本高昂、耗时且容易出错的工作。诸如 ImageNet（Deng等人，2009年）等常用数据集包含大量错误，即标签不正确或含义模糊的数据点（Beyer等人，2020年）。随着现代数据集规模不断扩大，对自动错误检测工具的需求也日益增加。

影响函数（IF）（Koh 和 Liang，2017年）及其变体（Charpiat 等人，2019年；Khanna 等人，2019年；Barshan 等人，2020年；Pruthi 等人，2020年）是用于估计一个数据点对另一个数据点影响的有力工具。研究人员利用影响函数的这一能力来设计或检测大规模数据集中的对抗性（Cohen 等人，2020年）、投毒（Koh 等人，2022年；Koh 和 Liang，2017年）以及错误的（Dau 等人，2022年）样本。直觉上，这些有害数据点通常会对其他数据点产生负面影响，而这种影响可以通过影响函数进行估计。

*共同第一作者

Basu 等人（2021年）通过实证观察发现，影响函数应用于深度神经网络（DNN）时具有不稳定性。随着网络变得更加复杂，影响估计的质量会下降。本文从实证和理论两方面解释影响函数的不稳定性。我们发现，当两个数据点属于不同类别时，影响函数得分噪声很大；而当两个数据点属于同一类别时，影响函数得分则稳定得多（第3节）。基于这一发现，我们提出了 IFs-class，这是影响函数的一种变体，它利用类别信息提高稳定性，同时不增加额外的计算成本。IFs-class 可以替代异常数据检测算法中的影响函数。在第4节中，我们比较了 IFs-class 与影响函数在错误检测问题上的表现。针对各种自然语言处理任务和数据集开展的实验，证实了 IFs-class 相较于影响函数的优势。

2 背景与相关工作

我们定义本文中使用的符号。设 $\mathbf{z} = (\mathbf{x}, \mathbf{y})$ 为一个数据点，其中 $\mathbf{x} \in \mathcal{X}$ 是输入，$\mathbf{y} \in \mathcal{Y}$ 是目标输出；$\mathcal{Z} = \{\mathbf{z}^{(i)}\}_{i=1}^{n}$ 为包含 n 个数据点的数据集；$\mathcal{Z}_{-i} = \mathcal{Z} \backslash \mathbf{z}^{(i)}$ 为移除 $\mathbf{z}^{(i)}$ 后的数据集；$f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$ 为参数为 $\boldsymbol{\theta}$的模型；$\mathcal{L}_{\mathcal{Z},\boldsymbol{\theta}} = \frac{1}{n}\sum_{i=1}^{n} \ell(f_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}), \mathbf{y}^{(i)}) = \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{z}^{(i)}; \boldsymbol{\theta})$ 为在 $\mathcal{Z}$ 上测量的 $f_{\boldsymbol{\theta}}$ 经验风险，其中 $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^{+}$ 为损失函数；$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{Z},\boldsymbol{\theta}}$ 以及 $\boldsymbol{\theta}_{-i} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{Z}_{-i},\boldsymbol{\theta}}$ 分别为在 $\mathcal{Z}$和 $\mathcal{Z}_{-i}$上训练的模型 $f_{\boldsymbol{\theta}}$ 的最优参数。在本文中，$f_{\boldsymbol{\theta}}$ 为一个深度网络，而 $\hat{\boldsymbol{\theta}}$ 通过在训练集 $\mathcal{Z}$上使用梯度下降训练 $f_{\boldsymbol{\theta}}$ 得到。

2.1 影响函数及其变体

数据点 $\mathbf{z}^{(i)}$ 对另一个数据点 $\mathbf{z}^{(j)}$ 的影响，定义为从训练集中移除$\mathbf{z}^{(j)}$时$\mathbf{z}^{(i)}$ 处损失的变化

$$s^{(ij)} = \ell(\mathbf{z}^{(j)}; \hat{\boldsymbol{\theta}}_{-i}) - \ell(\mathbf{z}^{(j)}; \hat{\boldsymbol{\theta}}) \qquad (1)$$

$s^{(ij)}$ 的绝对值衡量$\mathbf{z}^{(i)}$ 对$\mathbf{z}^{(j)}$的影响强度。 $s^{(ij)}$ 的符号表示影响方向。负的 $s^{(ij)}$ 意味着移除 $\mathbf{z}^{(i)}$ 会降低$\mathbf{z}^{(j)}$处的损失，也就是说$\mathbf{z}^{(i)}$对$\mathbf{z}^{(j)}$有害。 $s^{(ij)}$ 的方差较高，因为它取决于单个（任意的）数据点$\mathbf{z}^{(j)}$。为了更好地估计$\mathbf{z}^{(i)}$ 对整个数据分布的影响，研究人员会对参考集 $\mathcal{Z}'$ 中$\mathbf{z}^{(i)}$ 的影响分数取平均值

$$s^{(i)} = \frac{1}{|\mathcal{Z}'|} \sum_{\mathbf{z}^{(j)} \in \mathcal{Z}'} s^{(ij)} = \mathcal{L}_{\mathcal{Z}', \hat{\boldsymbol{\theta}}_{-i}} - \mathcal{L}_{\mathcal{Z}', \hat{\boldsymbol{\theta}}} \quad (2)$$

$s^{(i)}$ 是$\mathbf{z}^{(i)}$ 对参考集的影响 $\mathcal{Z}'$。$\mathcal{Z}'$ 可以是训练集的随机子集，也可以是留出数据集。朴素地计算 $s^{(ij)}$ 需要重新训练 $f_{\boldsymbol{\theta}}$ 于 $\mathcal{Z}_{-i}$。Koh 和 Liang (2017年) 提出了影响函数（IF），以便快速估计 $s^{(ij)}$ 而无需重新训练

$$\begin{aligned} s^{(ij)} &\approx IF(\mathbf{z}^{(i)}, \mathbf{z}^{(j)}) \\ &\approx \frac{1}{n} \nabla_{\hat{\boldsymbol{\theta}}} \ell(\mathbf{z}^{(i)}; \hat{\boldsymbol{\theta}})^\top H_{\hat{\boldsymbol{\theta}}}^{-1} \nabla_{\hat{\boldsymbol{\theta}}} \ell(\mathbf{z}^{(j)}; \hat{\boldsymbol{\theta}}) \end{aligned} \quad (3)$$

其中$H_{\hat{\boldsymbol{\theta}}} = \partial^2 \mathcal{L}_{\mathcal{Z},\, \hat{\boldsymbol{\theta}}} / \partial \boldsymbol{\theta}^2$ 是$\hat{\boldsymbol{\theta}}$处的海森矩阵。精确计算 $H^{-1}{}_{\hat{\boldsymbol{\theta}}}$ 对于现代网络来说难以实现。Koh 和 Liang (2017年) 开发了一种快速算法来估算 $H^{-1}{}_{\hat{\boldsymbol{\theta}}} \nabla_{\hat{\boldsymbol{\theta}}} \ell(\mathbf{z}^{(j)}; \hat{\boldsymbol{\theta}})$，并且仅使用最后一层参数的导数来提高算法速度。Charpiat 等人(2019年) 提出了梯度点积（GD）和梯度余弦相似度（梯度贡献，GC），作为更快的影响函数替代方法。Pruthi 等人(2020年) 认为，通过在整个训练过程中累积影响（TracIn），可以更好地近似影响。影响函数的公式总结于附录1中的A。

影响函数可以视为两个数据点的梯度之间相似度的度量。直观而言，有害样本的梯度与正常样本的梯度不同（图1）。

2.2 用于错误检测的影响函数

在错误检测问题中，我们需要检测标签错误的数据点。给定一个（可能含噪声的）数据集 $\mathcal{Z}$，我们需要按照数据点在$\mathcal{Z}$ 中出错的可能性对其排序。移除或更正错误可以提升在该数据集上训练的模型的性能和鲁棒性。

使用人工设计规则的传统错误检测算法（Chu 等人，2013）或简单统计量（Huang 和 He，2018年）难以扩展到深度学习数据集。 Cohen 等人（2020年）；Dau 等人 （2022年）使用影响函数检测深度学习数据集中的对抗样本和错误样本。 Dau 等人（2022年）使用影响函数衡量每个数据点 $\mathbf{z} \in \mathcal{Z}$ 对干净参考集的影响$\mathcal{Z}'$。其中的数据点 $\mathcal{Z}$ 会按其危害程度进行排序，排序依据是它们对 $\mathcal{Z}'$。危害最大的数据点会被重新经人工检查，或从 $\mathcal{Z}$ （算法 2，见附录 A）。本文聚焦于错误检测问题，但 IFs 和 IFs-class 也可用于检测其他类型的异常数据。

3 方法

3.1 动机

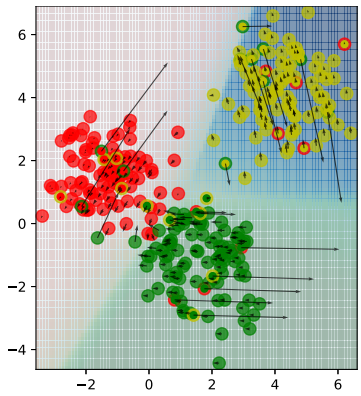


图 1：分类问题中的梯度模式。圆圈表示一个标签错误的数据点，内圈颜色代表原始（真实）类别，外圈颜色代表新的（噪声）类别。我们仅绘制梯度的前两个维度。实现细节及其他梯度维度请参见附录 C 。

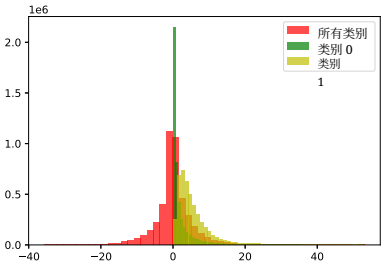


图 2：IMDB数据集上的 GD 分数分布。其他数据集上的结果见附录 C。

Basu 等人 （2021年）将影响函数的不稳定性归因于 DNN 的非凸性，以及泰勒展开和海森矩阵-向量乘积近似中的误差。本节将说明，DNN 的学习动态会使不同类别的样本彼此无关，并可能对彼此产生随机影响。

Pezeshkpour 等人（2021年）；Hanawa 等人（2021年）通过实证表明，使用最后一层梯度的影响函数与使用所有层梯度的影响函数表现相当或更好，而且不同变体的影响函数表现相似。因此，我们分析使用最后一层梯度的梯度下降行为，并将结果推广到其他影响函数。在图

1展示了一个三分类问题中多层感知机最后一层的梯度。图中，错误标注数据点的梯度幅度较大，且与真实类别中正确数据点的梯度方向相反。然而，错误标注数据点的梯度不一定与其他类别中正确数据点的梯度方向相反。此外，来自两个不同类别的两个数据点的梯度几乎相互垂直。我们得出以下观察结果：错误标注/正确标注的数据点通常对同一（真实）类别的数据点产生非常负面/正面的影响，但其对其他类别的影响则存在噪声且较小。

我们在真实世界数据集上验证了这一观察结果。（图 2）。我们计算来自两个不同类别的干净数据点对的梯度下降分数，并绘制分数分布。对每个类别中的数据点对重复这一过程。在二分类情况下，梯度下降分数几乎呈正态分布，并在 0 处有一个非常尖锐的峰值。这意味着，在许多情况下，一个类别中的干净数据点对另一个类别的数据点没有显著影响。而当它产生显著影响时，这种影响为正或为负的概率相同。相比之下，同一类别中数据点对的梯度下降分数几乎总是为正。一个干净数据点几乎必然会对同一类别中的干净数据点产生正向影响。

我们的理论分析表明，当两个数据点的标签不同时，梯度下降的符号取决于两个随机变量：特征内积的符号，以及损失相对于 l ogits 的梯度内积的符号。随着模型对两个数据点标签的置信度提高，梯度下降的幅度会非常迅速地减小。对 logits 或特征进行微小扰动，就可能翻转梯度下降的符号。相比之下，如果两个数据点具有相同的标签，那么梯度下降的符号仅取决于一个随机变量，即特征内积的符号；并且当模型的置信度提高时，梯度下降的幅度仍保持较大。数学细节参见附录。D。

算法 1 基于类别的影响函数错误检测。

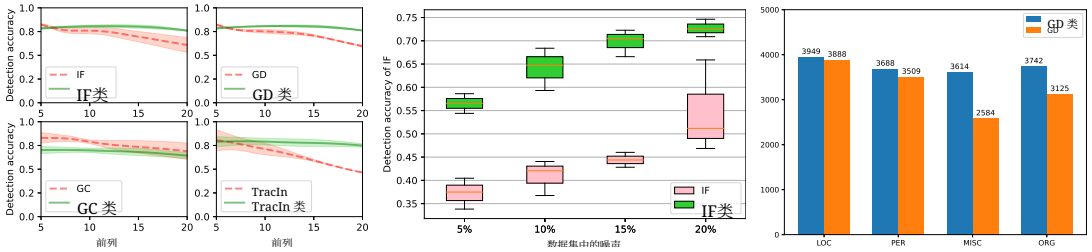
要求： **1:** $\mathcal{Z} = \{\mathbf{z}^{(i)}\}_{i=1}^{n}$**: 一个大型含噪数据集 2:** C**: 类别数量 3:** $\mathcal{Z}'_k = \{\mathbf{z}'^{(j_k)}\}_{j_k=1}^{m_k}$**: 来自类别** k**的干净数据 4:** $\mathcal{Z}' = \bigcup_{k=1}^{C} \mathcal{Z}'_k$**: 一个干净的参考数据集 5:** $f_{\hat{\theta}}$**: 在** $\mathcal{Z}$ **上预训练的深度模型 6: sim(**$\cdot,\cdot$**): 表 1 中的相似度度量 ˆ确保： ** $\mathcal{Z}$**: ** $\mathcal{Z}$ **中的数据点，按分数排序 7: 对** $\mathbf{z}^{(i)} \in \mathcal{Z}$ **执行 8: 对** $k = 1, ..., C$ **执行 9:** $s_k^{(i)} = \frac{1}{m_k} \sum_{j=1}^{m_k}$ **sim(**$\nabla_{\hat{\theta}} \ell(\mathbf{z}^{(i)})$ ∇ˆ$_{\theta} \ell(\mathbf{z}'^{(j_k)})$**) 10: 结束循环 11:** $s^{(i)} =$**min**$_k(s_k^{(i)})$ **12: 结束循环ˆ13:** $\mathcal{Z} =$ **sort(**$\mathcal{Z}$**,key** $= s,$ **reverse** **=True) ˆ14: 返回** $\mathcal{Z}$

3.2 基于类别的影响函数错误检测

我们的基于类别的影响函数错误检测方法如算法1所示。在第3.1节中，我们看到，错误数据点会对真实类别中的正确数据点产生非常强的负面影响，而正确数据点会对真实类别中的正确数据点产生正面影响。真实类别上的影响分数更能指示数据点的危害性，也更善于区分错误数据点和正确数据点。由于我们事先不知道 $\mathbf{z}^{(i)}$ 的真实类别，因此我们计算其在参考集中的每个类别上的影响分数 $\mathcal{Z}'$，并取这些影响分数中的最小值，作为衡量$\mathbf{z}^{(i)}$ 危害性的指标（第8-11行）。与原始影响函数不同，IFs-class 不会受到其他类别噪声的影响，因此具有更低的方差（附录4 中的A）。在附录A中，我们展示了我们的算法与基于影响函数的错误检测算法具有相同的计算复杂度。

4 实验

实验设置 我们在两项自然语言处理任务上评估 IFs-class 的错误检测性能：(1) IMDB（Maas 等人，2011）、SNLI（Bowman 等人，2015）和 BigCloneBench（Svajlenko 等人，2014）数据集上的文本分类；(2) CoNLL2003（Tjong Kim Sang 和 De Meul- der，2003）数据集上的命名实体识别。对于文本分类任务，我们检测标签错误的文本片段。对于命名实体识别任务，我们检测实体类型错误的词元。我们使用 BERT（Devlin等人，2019年）和



(a) (b) (c)

图 3：(a) SNLI 数据集上使用 $p = 20\%$ 时的错误检测准确率。(b) 在 IMDB 数据集上，IF 和 IF类在不同 p 值下的错误检测准确率。(c) GD 和 GD 类在 $p = 30\%$ $r = 30\%$ $q = 9\%$ 时检测到的错误命名实体识别词元数量，按错误词元的实体类型分组。

CodeBERT（Feng 等人，2020年）进行实验。实现细节见附录B。为了创建基准数据集 $\mathcal{Z}$，我们向上述数据集中注入随机噪声。对于文本分类数据集，我们随机选取 $p\%$ 的数据点，并将其标签更改为其他随机类别。对于 CoNLL-NER 数据集，我们随机选取$p\%$ 的句子，并将所选句子中 $r\%$的短语标签更改为同一类别。所选短语中的所有词元都被更改为相同的类别。参考集 $\mathcal{Z}'$ 通过从每个类别中随机选取 m_k 个干净数据点 $\mathcal{Z}$创建。模型在含噪数据集 $\mathcal{Z}$上训练。为了评估错误检测算法，我们从排序后的数据集 $q\%$ˆ中选取危害最大的 $\mathcal{Z}$ 个数据点，并检查所选数据点中有多少百分比确实是错误的。直观来说，增大q 可以让算法发现更多错误（提高召回率），但可能会降低检测准确率（降低精确率）。

结果与分析 由于所有数据集上的结果呈现相同的模式，我们在此报告具有代表性的结果，并将完整结果留待附录 C。

图 3(a)展示了 SNLI 数据集上的错误检测准确率，以及准确率随 q变化的情况。除了 GC 算法外，我们基于类别的算法比非类别版本具有更高的准确率和更低的方差。当q 增大时，IFs-class 的性能下降幅度没有 IFs 那么大。这证实了 IFs-class 的噪声比 IFs 小。类别信息未能提升 GC 的性能。为了理解这一点，我们重新考察相似度度量 $\mathrm{sim}(\cdot,\ \cdot)$。假设存在一些梯度非常大的干净数据点 $\mathbf{z}'^{(j)} \in \mathcal{Z}'$，其梯度为$\nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}'^{(j)})$。如果相似度度量没有对 $\nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}'^{(j)})$的范数进行归一化，那么 $\mathbf{z}'^{(j)}$ 将具有对影响分数影响最大。这些数据点是影响分数中的噪声的主要来源。梯度贡献会对两个梯度进行归一化，$\nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}^{(i)})$ 和 $\nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}'^{(j)})$，并有效去除这类噪声。然而，错误数据点的梯度往往大于正常数据点的梯度（图 1）。通过对两个梯度进行归一化，梯度贡献去除了错误数据点 $\nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}^{(i)})$梯度幅值中的宝贵信息。这会降低检测性能。图 3(a)显示，当 $q \geq 15\%$ 时，梯度贡献的性能低于其他基于类别的算法。在其他数据集上也观察到了类似趋势（附录 6、7、8 中的 C）。

图 3(b) 展示了噪声水平 p 从 5% 增至 20% 时检测准确率的变化。对于每个 p值，我们将 q 设为 p。我们的基于类别的影响分数显著提升了性能，并降低了方差。值得注意的是，当 p 增大时，由于错误数量增加，错误检测问题会变得更容易。因此，检测准确率通常会随着 p 增加，如图3(b)、9、10所示。

图 3(c)显示，在 CoNLL2003-NER 的所有实体类型上，GD 类的表现均优于梯度下降。GD 类与梯度下降之间的性能差距在其他和组织类别上更大。直观而言，人名可能是某个组织的名称，但反过来的可能性较小。因此，相较于将组织或其他标签更改为人名或地点标签，更难检测出人名或地点标签被更改为组织或其他标签。结果表明，IFs-class 在检测困难错误样本方面比影响函数更有效。

5 结论

本文研究影响函数，并找出其不稳定性的根源。我们从理论上解释了这些观察结果。我们提出了一种 IFs 的稳定变体，并以此来开-

发高性能错误检测算法。我们的发现有助于深入理解新型影响力估计器的开发，以及影响函数在下游任务中的应用。

局限性

我们的论文存在以下局限性

1. 我们的基于类别的影响分数无法提升 GC 算法的性能。尽管 GD、影响函数和 TracIn 的基于类别版本表现优于原始 GC，但我们旨在开发更强大的 GC 版本。根据第 4 节中的分析，我们认为，部分归一化的 GC 可能会有更好的表现。在部分 GC 中，我们仅对干净数据点 $\mathbf{z}'^{(j)}$ 的梯度进行归一化。这样可以去除由 $\|\nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}'^{(j)})\|$ 引入的噪声，同时保留有关 $\nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}^{(i)})$ 的范数的有价值信息。

伦理声明

我们的论文探讨了影响函数的一个理论层面，不对任何人群持有偏见。我们的研究结果不会对任何人群造成伤害。

参考文献

Elnaz Barshan、Marc-Etienne Brunet 和 Gintare Karolina Dziugaite。2020年。Relatif：通过相对影响识别具有解释力的训练样本。载于人工智能与统计国际会议，第1899–1909页。PMLR。

Samyadeep Basu、Phil Pope 和 Soheil Feizi。2021年。深度学习中的影响函数十分脆弱。载于国际学习表征会议。

Lucas Beyer、Olivier J. Hénaff、Alexander Kolesnikov、Xiaohua Zhai 和 Aäron van den Oord。2020年。我们是否已经解决了 ImageNet 问题？*CoRR*，abs/2006.07159。

Samuel R. Bowman、Gabor Angeli、Christopher Potts 和 Christopher D. Manning。2015年。用于自然语言推理学习的大型标注语料库。载于*2015*年自然语言处理经验方法会议论文集，第632–642页，葡萄牙里斯本。计算语言学协会。

Tom Brown、Benjamin Mann、Nick Ryder、Melanie Subbiah、Jared D Kaplan、Prafulla Dhariwal、Arvind Neelakantan、Pranav Shyam、Girish Sastry、Amanda Askell、Sandhini Agarwal、Ariel Herbert-Voss、Gretchen Krueger、Tom Henighan、Rewon Child、Aditya Ramesh、Daniel Ziegler、Jeffrey Wu、Clemens Winter、Chris Hesse、Mark Chen、Eric Sigler、Mateusz Litwin、Scott Gray、Benjamin Chess、Jack Clark、Christopher Berner、Sam McCandlish、Alec Radford、Ilya Sutskever 和 Dario Amodei。2020年。语言模型是少样本学习者。载于神经信息处理系统进展，第33卷，第1877–1901页。Curran Associates, Inc.

Guillaume Charpiat、Nicolas Girard、Loris Felardos 和 Yuliya Tarabalka。2019年。从神经网络视角看输入相似性。神经信息处理系统进展，32。

Xu Chu、Ihab F. Ilyas 和 Paolo Papotti。2013年。全息数据清理：将违规置于上下文中。载于*2013*年 *IEEE* 第*29*届国际数据工程会议（*ICDE*），第458–469页。

Gilad Cohen、Guillermo Sapiro 和 Raja Giryes。2020年。使用影响函数和最近邻检测对抗样本。载于*IEEE/CVF*计算机视觉与模式识别会议论文集，第14453–14462页。

Anh T. V. Dau、Nghi D. Q. Bui、Thang Nguyen-Duc 和 Hoang Thanh-Tung。2022年。探索使用数据影响方法检测源代码语料库中的噪声样本。载于 第*37*届*IEEE/ACM*国际自动化软件工程会议论文集。

Jia Deng、Wei Dong、Richard Socher、Li-Jia Li、Kai Li 和李飞飞。2009年。ImageNet：一个大规模分层图像数据库。载于 *2009*年*IEEE*计算机视觉与模式识别会议，第248–255页。IEEE。

Jacob Devlin、Ming-Wei Chang、Kenton Lee 和 Kristina Toutanova。2019年。BERT：用于语言理解的深度双向变换器预训练。载于*2019*年北美计算语言学协会分会会议论文集：人类语言技术，第*1*卷（长短论文），第4171–4186页，明尼阿波利斯，明尼苏达州。计算语言学协会。

Zhangyin Feng、Daya Guo、Duyu Tang、Nan Duan、Xi- aocheng Feng、Ming Gong、Linjun Shou、Bing Qin、Ting Liu、Daxin Jiang 和 Ming Zhou。2020年。Code- BERT：一种用于编程语言和自然语言的预训练模型。载于计算语言学协会 *EMNLP2020* 论文集，第 1536–1547 页，线上。计算语言学协会。

Daya Guo、Shuo Ren、Shuai Lu、Zhangyin Feng、Duyu Tang、Shujie Liu、Long Zhou、Nan Duan、Alexey Svyatkovskiy、Shengyu Fu 等。2020年。Graphcode- bert：利用数据流预训练代码表示。*arXiv* 预印本 *arXiv:2009.08366*。

Kazuaki Hanawa、ShoYokoi、Satoshi Hara 和 Ken- taro Inui。2021 年。基于相似度的解释的评估。载于国际学习表征会议。

Zhipeng Huang 和 Yeye He。2018年。Auto-detect：表格中的数据驱动错误检测。发表于*2018*年国际数据管理会议论文集，第 1377–1392 页。

Rajiv Khanna、Been Kim、Joydeep Ghosh 和 Sanmi Koyejo。2019年。使用费舍尔核解释黑箱预测。发表于第 *22* 届人工智能与统计学国际会议，第 3382–3390 页。PMLR。

Pang Wei Koh 和 Percy Liang。2017年。通过影响函数理解黑箱预测。发表于国际机器学习会议，第 1885–1894 页。PMLR。

Pang Wei Koh、Jacob Steinhardt 和 Percy Liang。2022年。更强的数据投毒攻击能够攻破数据净化防御。机器学习，111(1)：1–47。

Ilya Loshchilov 和 Frank Hutter。2019年。解耦权重衰减正则化。发表于国际学习表征会议。

ShuaiLu、Daya Guo、ShuoRen、Junjie Huang、Alexey Svyatkovskiy、Ambrosio Blanco、Colin Clement、Dawn Drain、Daxin Jiang、Duyu Tang、Ge Li、Li-dong Zhou、Linjun Shou、Long Zhou、Michele Tufano、MING GONG、Ming Zhou、Nan Duan、Neel

Sundaresan、Shao Kun Deng、Shengyu Fu 和 Shu-jie LIU。2021年。《CodeXGLUE：用于代码理解与生成的机器学习基准数据集》。发表于第三十五届神经信息处理系统大会数据集与基准赛道（第一轮）。

Andrew L. Maas、Raymond E. Daly、Peter T. Pham、Dan Huang、Andrew Y. Ng 和 Christopher Potts。2011年。用于情感分析的词向量学习。载于计算语言学协会第*49*届年会论文集：人类语言技术，第142–150页，美国俄勒冈州波特兰。计算语言学协会。

Pouya Pezeshkpour、Sarthak Jain、Byron Wallace 和 Sameer Singh。2021年。自然语言处理中实例归因方法的实证比较。载于计算语言学协会北美分会*2021*年会议论文集：人类语言技术，第967–975页，在线出版。计算语言学协会。

Garima Pruthi、Frederick Liu、Satyen Kale 和 Mukund Sundararajan。2020年。通过追踪梯度下降估计训练数据的影响。神经信息处理系统进展，33：19920–19930。

Jeffrey Svajlenko、Judith F. Islam、Iman Keivanloo、Chanchal K. Roy 和 Mohammad Mamun Mia。2014年。迈向大型数据集整理的跨项目代码克隆基准。载于*2014*年 *IEEE* 国际软件维护与演化会议，第476–480页。

Erik F. Tjong Kim Sang 和 Fien De Meulder。2003年。CoNLL-2003 共享任务简介：与语言无关的命名实体识别。载于*HLT-NAACL 2003* 第七届自然语言学习会议论文集，第142–147页。

Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Ł ukasz Kaiser 和 Illia Polosukhin。2017年。注意力就是你所需要的一切。载于神经信息处理系统进展，第30卷。Curran Associates, Inc.

A 附加算法与公式

表 1：影响函数及其变体。为清晰起见，我们省略常数因子 $1/n$。

IF	$\nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}^{(i)};\hat{\boldsymbol{\theta}})^\top H_{\hat{\boldsymbol{\theta}}}^{-1}\nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}^{(j)};\hat{\boldsymbol{\theta}})$
GD	$\left\langle \nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}^{(i)}), \nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}^{(j)}) \right\rangle$
GC	$\cos\left(\nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}^{(i)}), \nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}^{(j)})\right)$
TracIn	$\sum_{t=1}^{T}\eta_t \left\langle \nabla_{\boldsymbol{\theta}^{(t)}}\ell(\mathbf{z}^{(i)}), \nabla_{\boldsymbol{\theta}^{(t)}}\ell(\mathbf{z}^{(j)}) \right\rangle$

算法 2 基于影响函数的错误检测（Dau 等人，2022年）

要求：
1: $\mathcal{Z} = \{\mathbf{z}^{(i)}\}_{i=1}^{n}$：大型含噪数据集 2: $\mathcal{Z}' = \{\mathbf{z}'^{(j)}\}_{j=1}^{m}$：小型参考数据集 3: $f_{\hat{\boldsymbol{\theta}}}$：在……上预训练的深度模型
4: 相似度$(\cdot,\cdot)$：表 1 中的相似度度量

确保： 5: $\mathcal{Z}$：数据点，按分数 $\mathcal{Z}$ 排序 **对于**$\mathbf{z}^{(i)} \in \mathcal{Z}$ **执行**6: $s^{(i)} = \frac{1}{m}\sum_{j=1}^{m}$ 相似度$(\nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}^{(i)}), \nabla_{\hat{\boldsymbol{\theta}}}\ell(\mathbf{z}'^{(j)}))$ 7: **结束循环** 8: $\mathcal{Z}$ = 排序$(\mathcal{Z}$,键 $= \boldsymbol{s}$，升序 = 真$)$ 9: **返回** $\mathcal{Z}$

错误检测算法的计算复杂度

算法 1中的内层 for 循环计算 C 影响分数。它恰好调用评分函数 sim() $|\mathcal{Z}'| = m$ 次。算法 1 中内层 for 循环的复杂度 等于算法2中第 6 行的复杂度。因此，算法 1的复杂度等于算法 2的复杂度。

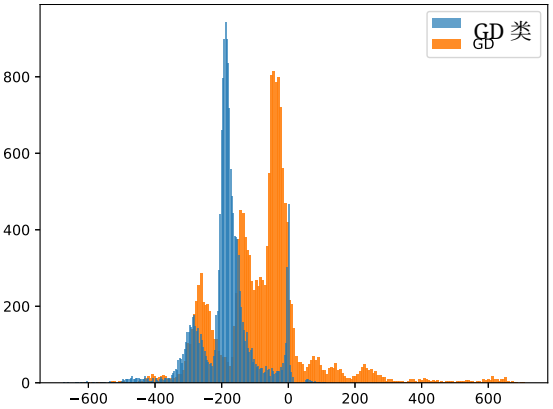


图 4：CoNLL2003数据集中错误词元的梯度下降和 GD 类分数分布。GD 类分数更集中，且大多为负值。梯度下降分数分布更分散，负值程度也较低。此外，相当一部分梯度下降分数大于 0，即梯度下降“认为”这些错误的数据点对 $\mathcal{Z}'$ 中的干净数据点具有正向影响。相比之下，GD 类分数更集中，且几乎总是为负值。这表明 GD 类明显优于梯度下降。

B 实现细节

B.1 实验设置

我们使用标准数据集和模型，采用 5 个不同的随机种子进行实验，并报告

均值和标准差。我们的实验在 Nvidia RTX 3090 上运行。模型使用 AdamW 优化器（Loshchilov 和 Hutter，2019年）进行训练，学习率为 $\eta = 5e-5$，使用交叉熵损失函数，批量大小为 16。我们采用验证集上分类准确率最高的轮次进行错误检测。

我们的源代码和指南已附在补充材料中。

B.2 数据集

IMDB(Maas 等人，2011) 该数据集包含来自互联网电影数据库（IMDb）网站的 50000 条评论。此任务是二元情感分析任务。数据集中的正面和负面评论数量相同。IMDB数据集分为训练集、验证集和测试集，大小分别为 17500、7500 和 25000。IMDB数据集可在此处获取：
https://ai.stanford.edu/~amaas/data/sentiment/

SNLI数据集（标准自然语言推理）（Bowman 等人，2015）由 57 万个人工标注的句子对组成，标签包括蕴含、矛盾和中性。我们将这些标签转换为数字。该数据集旨在作为评估文本表征系统的基准。该数据集可在此处获取：https://nlp.stanford.edu/projects/snli/。

BigCloneBench (Svajlenko 等人，2014）是一个大型代码克隆基准，包含来自 10 种不同功能的 6,000,000 多对真实克隆和 260,000 对虚假克隆。任务是预测两段代码的语义是否相同。该数据集常用于代码语言模型（Feng 等人，2020年；Lu 等人，2021年；Guo 等人，2020年）。该数据集可在以下网址获取：
https://github.com/clonebench/BigCloneBench

CoNLL2003 (Tjong Kim Sang和De Meulder，2003）是命名实体识别模型研究中最具影响力的语料库之一。许多出版物（包括许多里程碑式的研究成果）都将该语料库用作命名实体识别任务的真实标注来源。数据包含两种语言：英语和德语。本文使用 CoNLL2003 英语数据集。训练集、验证集和测试集分别包含 14,987、3,466和3,684个句子，对应203,621、51,362和46,435个词元。该数据集可在此获取：
https://www.clips.uantwerpen.be/conll2003/ner/

B.3 模型

BERT(Devlin等人，2019年）是 Bidirectional Encoder Representations from Transformer（来自 Transformer 的双向编码器表征）的缩写，基于 Transformer。本文使用的 BERT 模型经过预训练，可用于自然语言处理任务。我们在 IMDB 和 SNLI 数据集上使用 BERT。同时，我们也将 BERT 模型用于CoNLL2003数据集上的命名实体识别问题。

CodeBERT(Feng 等人，2020年）是一种用于编程语言和自然语言的双模态预训练模型。我们使用 CodeBERT 处理 BigCloneBench 数据集。

C 其他结果

C.1 三分类实验

我们使用随机梯度下降训练一个多层感知机，包含 2 个输入神经元、第一隐藏层中的 100 个隐藏神经元、第二隐藏层中的 2 个隐藏神经元以及 3 个输出神经元，共训练 1000 个训练轮次。激活函数为 LeakyReLU，学习率为 $\eta = 1e-3$。最后一层包含 6 个参数，这些参数组织成一个 3×2 矩阵。损失相对于最后一层参数的梯度也组织成一个 3×2 矩阵。我们在 3 个子图中展示梯度矩阵的 3 行（图5）。

C.2 IMDB、SNLI、BigCloneBench 和 CoNLL2003 上的结果

为确保我们的基于类别的算法与算法2之间进行公平比较，我们对两种算法使用相同的参考数据集 $\mathcal{Z}'$。该参考数据集 $\mathcal{Z}'$ 由 C 个类别组成。对于$C = 2$ IMDB数据集，我们使用 $C = 3$ SNLI 数据集使用 $C = 2$，BigCloneBench数据集使用 $C = 5$，CoNLL2003-NER数据集使用 C。对于每个 $m_k = 50k = 1$、.....、C类别，我们随机选择 $\mathcal{Z}'$个干净数据点以构成 m_k。我们尝试将从 10 调整到 1000，观察到性能没有显著变化。

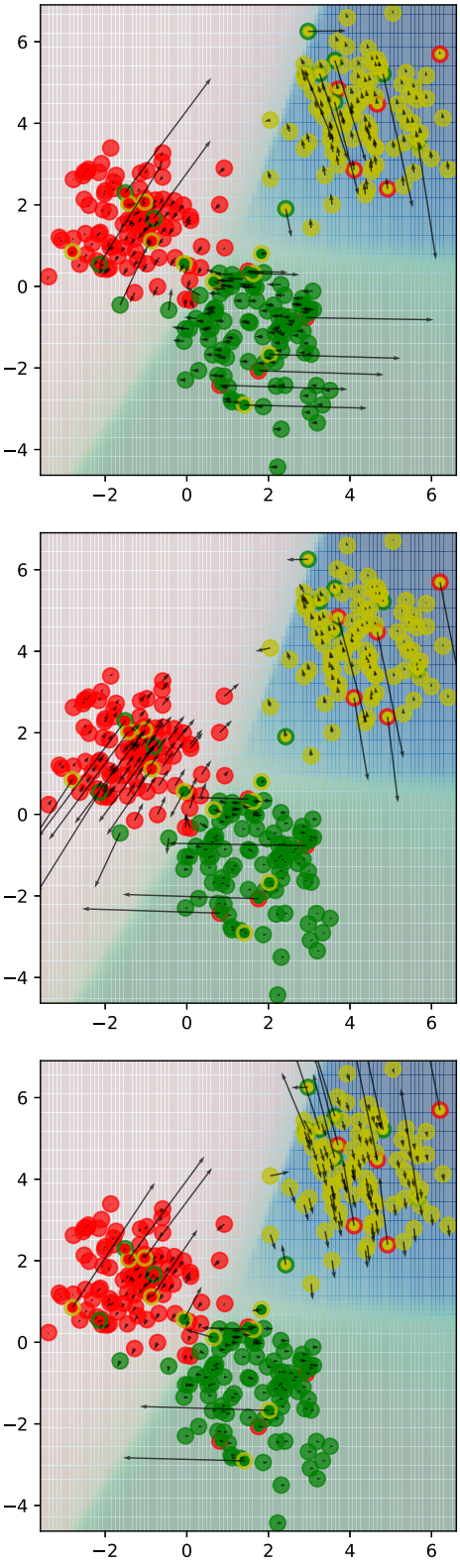


图5：分类问题上的梯度模式。每个子图展示梯度的两个维度。顶部子图展示梯度的第1和第2个维度。中间子图展示梯度的第3和第4个维度。底部子图展示梯度的第5和第6个维度。

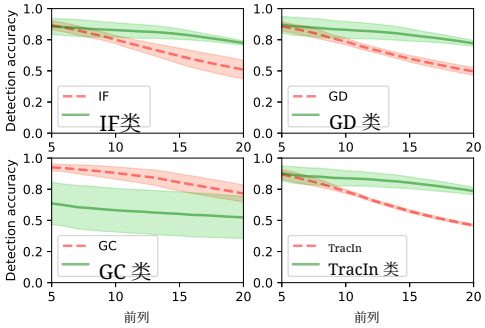


图6：IMDB数据集上的错误检测准确率，数值为 $p = 20\%$。

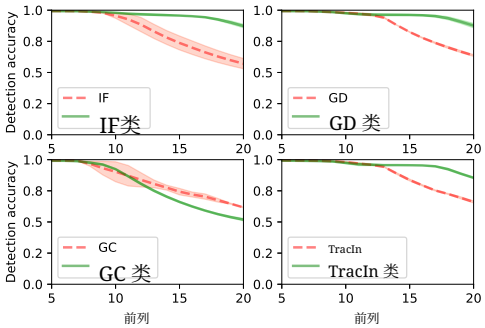


图7：BigCloneBench数据集上的错误检测准确率为 $p = 20\%$。

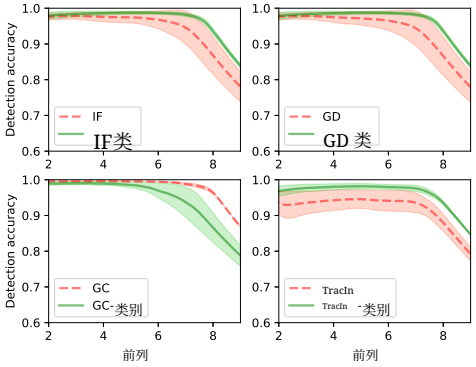


图8：CoNLL2003数据集上的错误检测准确率为 $p = 30\%$和 $r = 30\%$

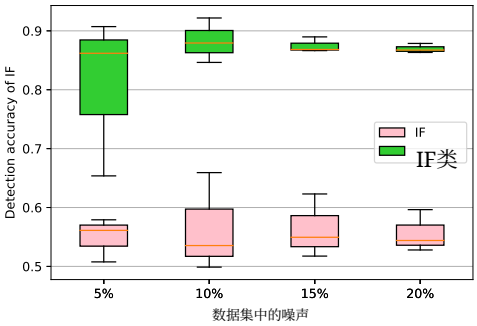


图 9：随着噪声水平变化，BigCloneBench数据集上的错误检测准确率变化。

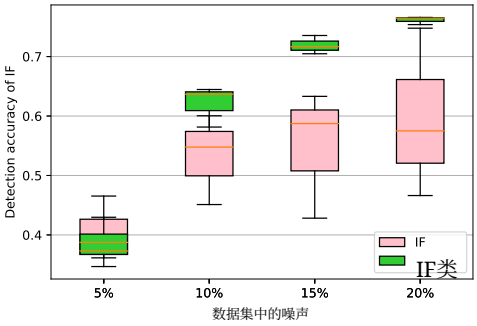


图 10：随着噪声水平变化，SNLI 数据集上的错误检测准确率变化。

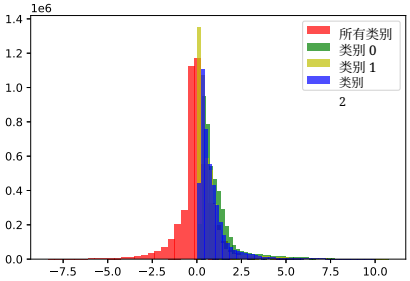


图 11：SNLI 数据集上的 GD 分数分布。

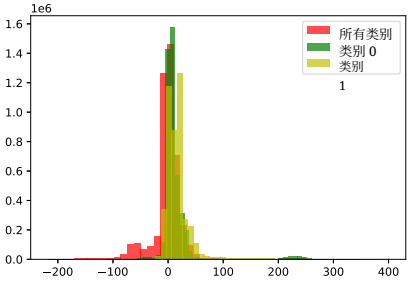


图 12：BigCloneBench 数据集上的 GD 分数分布。

D 第 3 节中观察结果的解释3

考虑一个采用交叉熵损失函数的分类问题

$$\ell(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{i=1}^{d_y} y_i \log \hat{y}_i$$

其中 d_y 是类别数。设$\mathbf{z} = (\mathbf{x},\mathbf{y})$为一个标签为 k 的数据点，即 $y_k = 1,\ y_i = 0 \ \forall\ i \neq k$。模型 $f_{\boldsymbol{\theta}}$ 是一个深度网络，其最后一层的参数为 $W \in \mathbb{R}^{d_y \times d_h}$，其中 d_h 是隐藏神经元的数量。设$\mathbf{u} \in \mathbb{R}^{d_h}$ 为倒数第二层的激活值。输出的计算方式如下

$$\mathbf{a} = W\mathbf{u}$$
$$\hat{\mathbf{y}} = \delta(\mathbf{a})$$

其中 δ 是 softmax 输出函数。在$\mathbf{z}$ 处，损失关于 W 的导数为

$$\frac{\partial \ell(\mathbf{z})}{\partial W} = \nabla_{\mathbf{a}} \ell(\mathbf{z})\ \mathbf{u}^\top \tag{4}$$
$$= \begin{bmatrix} \nabla_{\mathbf{a}} \ell(\mathbf{z})_1 \mathbf{u}^\top \\ \vdots \\ \nabla_{\mathbf{a}} \ell(\mathbf{z})_{d_y} \mathbf{u}^\top \end{bmatrix} \tag{5}$$

梯度 $\nabla_{\mathbf{a}} \ell(\mathbf{z})$为

$$(\nabla_{\mathbf{a}} \ell)^\top = \frac{\partial \ell}{\partial \mathbf{a}} \tag{6}$$
$$= \frac{\partial \ell}{\partial \hat{\mathbf{y}}} \frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{a}} \tag{7}$$
$$= \begin{bmatrix} \frac{\partial \ell}{\partial \hat{y}_1} & \cdots & \frac{\partial \ell}{\partial \hat{y}_k} & \cdots & \frac{\partial \ell}{\partial \hat{y}_{d_y}} \end{bmatrix} \times \begin{bmatrix} \frac{\partial \hat{y}_1}{\partial a_1} & \frac{\partial \hat{y}_1}{\partial a_2} & \cdots & \frac{\partial \hat{y}_1}{\partial a_{d_h}} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial \hat{y}_k}{\partial a_1} & \frac{\partial \hat{y}_k}{\partial a_2} & \cdots & \frac{\partial \hat{y}_k}{\partial a_{d_h}} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial \hat{y}_{d_y}}{\partial a_1} & \frac{\partial \hat{y}_{d_y}}{\partial a_2} & \cdots & \frac{\partial \hat{y}_{d_y}}{\partial a_{d_h}} \end{bmatrix} \tag{8}$$
$$= \begin{bmatrix} \frac{\partial \ell}{\partial \hat{y}_k} \frac{\partial \hat{y}_k}{\partial a_1} & \cdots & \frac{\partial \ell}{\partial \hat{y}_k} \frac{\partial \hat{y}_k}{\partial a_k} & \cdots & \frac{\partial \ell}{\partial \hat{y}_k} \frac{\partial \hat{y}_k}{\partial a_{d_h}} \end{bmatrix} \tag{9}$$

我们从公式8 推导到公式9时，使用以下事实

$$\frac{\partial \ell}{\partial \hat{y}_i} = \begin{cases} 0 \text{ if } i \neq k \\ \frac{1}{\hat{y}_i} \text{ if } i = k \end{cases}$$

我们还有

$$\frac{\partial \hat{y}_k}{\partial a_i} = \begin{cases} \hat{y}_k(1 - \hat{y}_k) \text{ if } i = k \\ -\hat{y}_k \hat{y}_i \text{ if } i \neq k \end{cases}$$

将其代入式 9 可得

$$\nabla_{\mathbf{a}} \ell = \begin{bmatrix} -\hat{y}_1 \\ \vdots \\ 1 - \hat{y}_k \\ \vdots \\ -\hat{y}_{d_y} \end{bmatrix}$$

因为 $1 - \hat{} y_k = \sum_{j \neq k} \hat{} y_j$，$1 - \hat{} y_k$ 通常远大于 $\hat{} y_j$ 将其代入公式5，可见第 k行的幅值远大于其他行。我们还注意到，k行的 W 更新方向与其他行的更新方向相反。

我们来考察两个数据点的梯度内积：$\mathbf{z}$ 和$\mathbf{z}'$，其标签分别为 k 和 k'。我们先考虑 $k' \neq k$ 的情况。

$$\text{vec}\left(\frac{\partial \ell(\mathbf{z})}{\partial W}\right)^\top \text{vec}\left(\frac{\partial \ell(\mathbf{z}')}{\partial W}\right) = (\nabla_{\mathbf{a}} \ell^\top \nabla_{\mathbf{a}'} \ell)(\mathbf{u}^\top \mathbf{u}') \tag{10}$$

直观而言，乘积 $\nabla_{\mathbf{a}} \ell^\top \nabla_{\mathbf{a}'} \ell$ 很小，因为较大的元素 $\nabla_{\mathbf{a}} \ell_k = 1 - \hat{} y_k$ 与较小的元素 $\nabla_{\mathbf{a}'} \ell_k = \hat{} y'_k$ 相乘，而较大的元素 $\nabla_{\mathbf{a}'} \ell_{k'} = 1 - \hat{} y'_{k'}$ 与较小的元素 $\nabla_{\mathbf{a}} \ell_{k'} = \hat{} y_{k'}$相乘。具体来说，假设 $\hat{} y_k = \alpha \approx 1$ 和 $\hat{} y_i = \frac{1-\alpha}{d_y - 1} = \beta$对于$i \neq k$成立。对于 $\hat{} \mathbf{y}'$，我们也作相同假设。

$$\nabla_{\mathbf{a}} \ell^\top \nabla_{\mathbf{a}'} \ell = (\hat{y}_k - 1)\hat{y}'_k + (\hat{y}'_{k'} - 1)\hat{y}_{k'} + \sum_{i=1, i \neq k, k'}^{d_y} \hat{y}_i \hat{y}'_i$$
$$= (d_y - 2)\beta^2 - 2(d_y - 1)\beta^2$$
$$= -d_y \beta^2$$
$$= -\frac{d_y (1 - \alpha)^2}{(d_y - 1)^2} \tag{11}$$

$\alpha \approx 1$ 意味着 $1 - \alpha \approx 0$ 并且 $\beta \approx 0$。公式 11 意味着，随着模型对 $\mathbf{z}$ 和 $\mathbf{z}'$ 的标签越来越有信心，乘积 $\nabla_{\mathbf{a}} \ell^\top \nabla_{\mathbf{a}'} \ell$ 会以二次速率趋近于 0。这意味着，随着训练的进行，来自不同类别的数据点会变得越来越独立。来自不同类别的数据点的梯度也会变得越来越垂直。

梯度乘积的符号取决于 $\nabla_{\mathbf{a}}\ell^{\top}\nabla_{\mathbf{a}'}\ell$ 和 $\mathbf{u}^{\top}\mathbf{u}'$的符号。$\nabla_{\mathbf{a}}\ell^{\top}\nabla_{\mathbf{a}'}\ell$ 和 $\mathbf{u}^{\top}\mathbf{u}'$ 的符号是随机变量，取决于特征 $\mathbf{u}$ 和 $\mathbf{u}'$ 中的噪声以及权重矩阵 W。如果模型 $f_{\boldsymbol{\theta}}$ 无法学到良好的输入表征，那么特征 $\mathbf{u}$ 和 $\mathbf{u}^{\top}\mathbf{u}'$ 的符号可能会有很大噪声。若 $\mathbf{z}$ 和 $\mathbf{z}'$ 来自不同类别，$\text{sign}(\mathbf{u}^{\top}\mathbf{u}')$ 的噪声会更大。由于 $\left|\nabla_{\mathbf{a}}\ell^{\top}\nabla_{\mathbf{a}'}\ell\right|$ 很小，logits $\mathbf{a}$ 和 $\mathbf{a}'$ 中的微小噪声就可能翻转 $\nabla_{\mathbf{a}}\ell^{\top}\nabla_{\mathbf{a}'}\ell$的符号并改变影响方向。

现在考虑 $k' = k$ 的情况。当$k' = k$时，$\nabla_{\mathbf{a}}\ell^{\top}\nabla_{\mathbf{a}'}\ell$ 始终为正。梯度乘积的符号仅取决于 $\mathbf{u}^{\top}\mathbf{u}'$。这解释了为什么来自同一类别的数据点的梯度乘积噪声小得多，并且几乎总是为正。

此外，$\nabla_{\mathbf{a}}\ell^{\top}\nabla_{\mathbf{a}'}\ell$ 的幅值大于 $k' \neq k$ 情况下的幅值，因为较大的元素 $1 - \hat{}y_k$ 会与较大的元素 $1 - \hat{}y'_k$ 相乘。更具体地说，在与情况 $k' \neq k$相同的假设下，我们有

$$\begin{aligned}\nabla_{\mathbf{a}}\ell^{\top}\nabla_{\mathbf{a}'}\ell &= (1-\hat{y}_k)(1-\hat{y}'_k) + \sum_{i=1, i\neq k}^{d_y} \hat{y}_i\hat{y}'_i \\ &= (1-\alpha)^2 + (d_y - 1)\beta^2 \end{aligned} \tag{12}$$

根据公式12，我们可以看到，当 $k' = k$时，$\nabla_{\mathbf{a}}\ell^{\top}\nabla_{\mathbf{a}'}\ell$的幅值约为 d_y 在 $k' \neq k$时的倍。