Міністерство освіти і науки, молоді та спорту України
Львівський національний університет імені Івана Франка
Факультет прикладної математики та інформатики
Кафедра обчислювальної математики

Курсова робота

на тему:

"Розробка алгоритмів захисту від атак на глибокі нейронні мережі"

Виконав:
студент IV курсу групи ПМп-41
напрямку підготовки (спеціальності)
113 – "Прикладна математика"
Бугрій Б.О.

Науковий керівник:
доц. Музичук Ю.М.

Львів - 2021

Зміст

Вступ

Сьогодні розумні системи штучного інтелекту є невід'ємною частиною життєдіяльності суспільства. Завдяки таким системам людству вдалося досягти значних результатів у багатьох сферах та галузях, таких як, наприклад, медицина, програмна інженерія, машинобудування та робототехніка.

Зараз найбільш поширеним типом алгоритмів машинного навчання є глибокі нейронні мережі, які здатні знаходити закономірності у великих масивах даних, а результати їх роботи часто перевершують людей. Проте, як показали нещодавні дослідження [1], такі алгоритми часто використовують антиінтуїтивні, в порівнянні зі смисловим значенням, закономірності стосовно певних характеристик. Через це нейронні мережі є вразливими до різного роду зловмисних втручань, які можуть призвести до невірних результатів.

Моделі, які показують відмінні результати на звичайних даних, можуть бути легко ошуканими зразками, які лише трохи відрізняються від правильно класифікованих прикладів. Це розкриває фундаментальні недоліки в алгоритмах машинного навчання, які можуть бути використані зловмисниками для заподіяння шкоди, що може вплинути на безпеку технологічних процесів людської життєдіяльності і призвести до катастроф великого масштабу.

Природним є бажання розробити архітектури та алгоритми тренування нейронних мереж, які зменшують ризики таких атак та є стійкими до зловмисних збурень. Постає питання, які саме фактори впливають на стійкість моделі, що робіть її більш вразливою до атак, а що ні.

... TODO

???

Щоб ефективно захиститись від можливих загроз, потрібно знати слабкі місця алгоритмів та стратегії нападу, що використовують зловмисники. Саме тому ми розглянемо деякі підходи до атак на нейронні мережі, знання яких дозволить покращити системи машинного навчання, зробити їх більш стійким до зловмисних втручань. Також необхідним буде ввести метрики стійкості мереж, щоб мати можливість об'єктивно порівняти отримані результати та обрати оптимальні з них.

У цій роботі ми розглянемо два абсолютно різні підходи до побудови захисту проти ошукуючих атак та спробуємо поєднати їх для отримання бажаного результату.

... TODO

1 Опис проблеми

Для повного розуміння поставленої мети, у цьому розділі ми формалізуємо постановку задачі побудови нейронних мереж, стійких до ошукуючих атак. Також ми опишемо пов'язані терміни та поняття, типи захистів та іншу інформацію, яку потім використаємо в наступних розділах.

1.1 Постановка задачі

Нехай система машинного навчання M на основі зразків $x \in S$ робить передбачення y. Тут S – множина всеможливих зразків-зображень з предметної області, які допустимі для використання моделлю M.

Означення 1.1 *Зразок* $x^{adv} = x + \tau$, $x^{adv} \in S$ *називається ошукуючим якщо для достатньо малих збурень* τ *виконуються такі умови:*

$$M(x) = y_{true} \tag{1.1}$$

$$M(x^{adv}) \neq y_{true} \tag{1.2}$$

де y_{true} *- правильне передбачення.*

Нашою метою є побудова максимально ефективної моделі машинного навчання, яка буде менш вразливою до такого типу ошукуючих зразків. Задачу захисту моделі від ошукуючої атаки можна формалізувати наступним чином.

Нехай $S^{adv}(M) \subset S$ – множина ошукуючих зразків для моделі M. Необхідно знайти модель M' яка є, в певному сенсі, модифікацією оригінальної моделі, таку, що

$$S^{adv}(M') = \emptyset. \tag{1.3}$$

Мається на увазі, що для моделі M' неможливо створити ошукуючі зразки, тобто вона є невразливою до атак.

Такий "ідеальний" випадок є практично неможливим, тому ми будемо використовувати пом'якшене формулювання, а саме вимагатимемо, щоб для моделі-образа задовільнялася умова

$$n(S^{adv}(M')) < n(S^{adv}(M)) \tag{1.4}$$

де $n(S)$ –кількість елементів в множині S. Іншими словами, нашою метою є побудова моделі, більш стійкої за оригінал.

1.2 Альтернативна постановка задачі

[TODO] додати або викинути

Модель машинного навчання вважається стійкою до ошукуючих зразків якщо результат передбачення не зміниться при невеликих збуреннях ...

1.3 Типологія захисту

Зважаючи на те, що існує багато абсолютно різних варіантів та підходів до захисту нейронних мереж від атак, доцільним є ввести певну класифікацію.

Першим спадає на думку поділ *за типом атак*, проти яких ми намагаємось побудувати захист. Тоді всі стратегії захисту можемо розподілити на три групи.

- **Захист від ошукуючих атак.** В основу ошукуючих атак покладена модифікація вхідних даних таким чином, щоб модель машинного навчання не могла розпізнати їх вірно. Така вразливість спричинена тим, що більшість методів машинного навчання розроблені для роботи над конкретними наборами проблем, набори даних для навчання та тестування яких генеруються з одного і того ж статистичного розподілу. Коли ці моделі застосовуються в реальному світі, супротивники можуть надавати зразки, що порушують це статистичне припущення, що призводить до компрометації результатів. Існує багато різноманітних алгоритмів і підходів до побудови захисту від такого типу атак і деякі з них ми розглянемо у цій роботі.

- **Захист від викрадення.** Розробка високопродуктивних глибоких нейронних мереж зазвичай потребує багато зусиль, часу та грошей. Саме тому зловмисники шукають шляхи як заволодіти цінною інформацією, яка допоможе відтворити модель за умов обмеженого доступу. Наприклад, вони можуть використати результати її передбачень для генерації власного тренувального набору та аналізу градієнтів. Завдання захисника у такому випадку – зберегти як більше інформації в таємниці, щоб не надати супротивнику можливість побудувати аналогічну модель.

- **Захист від отруєння.** Отруєння моделі зазвичай передбачає забруднення навчальних даних на яких тренується модель. Воно вважається атакою на цілісність, оскільки втручання в навчальні дані впливає на здатність моделі видавати правильні прогнози через зсув границі рішень. Найпоширеніші заходи захисту від таких атак це попередня обробка даних та обмеження доступу до них.

Оскільки нам доступні різні підходи до побудови захисту, варто також виділити ще одну вертикаль класифікації - *за стратегією захисту*. Тут можемо виділити такі стратегії, як:

- **Модифікація архітектури моделі та процесу тренування.** Ця стратегія включає в себе зміну кількості шарів та нейронів у них або додавання до вже наявної моделі нових шарів специфічної природи,

які, наприклад, можуть відсіювати зайвий шум, зменшувати розмірність зразка чи додавати до результатів деякі випадкові величини, щоб зробити важчим завдання пошуку ошукуючих градієнтів. Процес і алгоритм тренування також може бути зміненим, щоб отримати модель, стійкішу до атак.

- **Генерація специфічного тренувального набору.** Додавання спеціально підготованих зразків до тренувального набору також може покращити стійкість моделі до атак. Одним із очевидних підходів є додавання до множини вхідних даних власноруч згенерованих ошукуючих зразків, щоб охопити більшу предметну область. Деякі науковці також розглядають підходи до модифікації міток для вже існуючих екземплярів з метою кодування в них додаткової інформації.

- **Створення захисної оболонки.** Уже готова модель машинного навчання може бути поміщена в обмежене середовище, яке містить в собі засоби для захисту від атаки. Тобто оболонка моделі - це проксі-середовище, яке може по різному обмежувати комунікацію моделі. Наприклад, додаткова нейронна мережа може бути застосована для визначення шкідливих вхідних даних та недопущення таких даних до моделі. Захисник також може створити обмеження на кількість запитів, допустимих від одного користувача, та часовий інтервал між ними. Інформація про надмірну активність окремих користувачів може бути використана для ідентифікації атак.

Важливим фактором є те, що до одної нейронної мережі можна одночасно застосовувати різні типи захисту, що в деяких випадках може сприяти покращенню стійкості мережі до атак. Але потрібно зважати на те, що деякі стратегії захисту незначно погіршують результати моделі або збільшують затрати на її тренування та підтримку. Тому застосування надмірної кількості стратегій захисту може зробити модель непридатною для використання. У даній роботі ми сконцентруємося на дослідженні захисту від ошукуючих атак застосувавши різні підходи та стратегії, щоб спробувати досягти оптимальних результатів.

2 Глибокі нейронні мережі

3 Визначення стійкості

Під час проведення досліджень важливою є можливість порівняти отримані результати щоб визначити ефективність того чи іншого методу та вибрати підхід до тренування та застосування моделі, який робить її більш стійкою до ошукуючих зразків.

Отже, потрібно якимсь чином виміряти стійкість моделі. Тут варто згадати, якими метриками користуються зловмисники. У нашій попередній роботі [13] ми вже згадували, що під час атак основною метою є знайти зразок x^{adv} який є найближчим до оригінального зразка x. Щоб визначити "близькість" ошукуючих зразків зловмисники часто використовують норму L_p, де $1 \leq p < \infty$. Вона має вигляд

$$\|x - \tilde{x}\|_p = \left(\sum_{i=1}^{n} (x_i - \tilde{x}_i)^p \right)^{\frac{1}{p}} \tag{3.1}$$

Природнім є обмежувати розмір шуму, який зловмисники накладають на оригінальні зображення. Ці обмеження також часто є залежними від вищезгаданої норми. Ба більше, як ми вже згадували в нашій попередній роботі, деякі дослідники розглядають проблему мімімізації шуму як частину задачі створення ошукуючих зразків [12].

Значення (3.1) буде дуже малим, якщо шум складається з невеликих змін для багатьох пікселів, що, загалом, свідчить про те що людині складно буде відрізнити ошукуючий зразок від оригнального. З цього припущення випливає, що якщо шум τ є достатньо великим, то ошукуючий зразок було важче створити і його легше можна виявити. Далі розглянемо можливі варіанти визначення стійкості проти такого ошукуючого зразка.

Нехай $B_p(r) := \{\tau \in \mathbb{R}^n : \|\tau\|_p \leq r\}$ це куля, утворена на основі p -норми з радіусом r. Для заданої моделі класифікації, M, та фіксованого вхідного зразка, $x \in \mathbb{R}^n$, зловмисники можуть успішно створити ошукуючий зразок x^{adv} з розміром шуму L для обраної p-норми якщо їм вдастся знайти $\tau \in B_p(L)$ таке, що $M(x + \tau) \neq M(x)$. Узагальнивши, можна сказати що атакуючі намагаються знайти якнайменше τ яке змінить результат класифікації.

Інтуітивно зрозуміло, що модель можна розглядати як стійку, в певному сенсі, до ошукуючих зразків, якщо результати її передбачень є нечутливими до малих змін будь-якого прийнятного вхідного зразка. Важко формально визначити, які зразки можна вважати прийнятними. Оскільки перевірка продуктивності моделі проводиться на тестувальних даних, які не використовують під час тренування моделі, то такий набір можна застосувати і як початкові дані для оцінки стійкості.

Означення 3.1 *Модель M називається стійкою до атак за нормою L_p на*

заданому зразку x *коли* $M(x) = M(x+\tau)\ \ \forall\tau \in B_p(L)$. *Якщо* M *це НМ для багатокласової класифікації, то це рівняння еквівалентне до:*

$$\forall\tau \in B_p(L) : \hat{y}_k(x+\tau) > \max_{i:i\neq k} \hat{y}_i(x+\tau) \tag{3.2}$$

де $k := M(x)$.

[TODO] переглянути позначення.

Тобто якщо модель є певною мірою стійкою, то малі зміни вхідного зразка не змінюють оцінки настільки, щоб змінити результат передбачення. Можемо узагальнити даний принцип на весь тренувальний набір для визначення стійкості моделі використавши середню відстань від оригінальних зразків до ошукуючих .

Означення 3.2 *Нехай* M *– НМ для багатокласової класифікації,* $x \in X_{test}$ *– тестувальний набір,* $x^{adv} \in S^{adv}(M, X_{test})$ *– ошукуючі зразки, які відповідають прикладам з тестового набору. Тоді стійкість моделі на обраному наборі даних та з використанням відповідної норми визначимо як*

$$r_p(M, X_{test}) := \frac{\sum_{i=1}^{n_{test}} \|x_i - x_i^{adv}\|_p}{n_{test}} \tag{3.3}$$

де $n_{test} = n(X_{test})$. *Також можемо визначити, що* $r_p(M) \approx r_p(M, X_{test})$ *при* $n(X_{test}) \to \infty$.

Метрику $r_p(M, X_{test})$ зручно використовувати для оцінки стійкості моделі. Вона дає можливість чисельно порівняти роботу до та після застосування одної з стратегій захисту та при застосуванні різних алгоритмів.

4 Огляд атак на моделі машинного навчання

[TODO]

У своїй попередній роботі ...

Нейронні мережі є особливо чутливими до атак на білу скриньку. Отримавши доступ до всіх параметрів, у зловмисників з'являється можливість обчислити точні похідні для функцій (у випадку диференційовності моделі), за допомогою яких виконується класифікація. Тоді під час атаки можна застосувати різні чисельні методи для вирішення конкретної проблеми, що дозволяє досягти високої ефективності згенерованих ошукуючих зразків.

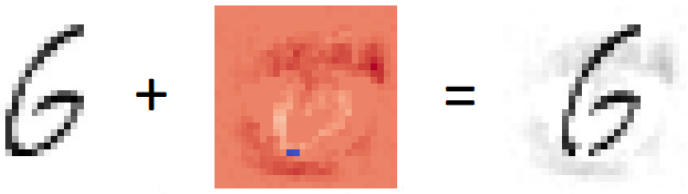

Рисунок 1. Початкове зображення класифікується нейронною мережею як "шість". Після додавання до нього певного шуму НМ класифікує його як "п'ять".

4.1 Швидкий градієнтний спуск

Метод швидкого градієнтного спуску є одним з найпростіших методів генерації ошукуючих прикладів. Його застосовують для атак на різні системи машинного навчання, зокрема й на нейронні мережі [8].

Основною метою методу є максимізація функції витрат J, розглядаючи її як $J(x, y_{true})$, зафіксувавши ваги нейронної мережі. В такому випадку потрібно знайти x^{adv} таке, що виконується нерівність:

$$J\left(x^{adv}, y_{true}\right) > J\left(x, y_{true}\right) \tag{1}$$

Тоді, згідно з [9], ми зможемо знайти розв'язок поставленої задачі. Пошук ошукуючих зразків здійснюємо за формулою:

$$x^{adv} = x + \epsilon sign\left(\nabla_x J\left(x, y_{true}\right)\right) \tag{2}$$

де ϵ - додатний параметр, який називатимемо *розміром кроку*. Напрямок $sign\left(\nabla_x J\left(x, y_{true}\right)\right)$ є *напрямком зростання* для функції J.

Варто зауважити, що метод не гарантує знаходження ошукуючого зразка, бо нерівність (1) свідчить лише про збільшення функції правдоподібності. Тому в загальному випадку доцільно використовувати ітераційну форму методу. Також можна застосовувати інші методи, такі як [12], для того, щоб зменшити розмір шуму τ.

4.2 Оптимізація нульового порядку

Більшість реальних систем, які використовують нейронні мережі, не розголошують своїх конфігурацій (як от структура мережі і її ваги), тому атаки, на кшталт швидкого градієнтного спуску не так просто застосувати на практиці. В деяких сценаріях зловмисники мають можливість використовувати модель для отримання передбачень на основі власних зразків (як от, наприклад, при використанні програмних інтерфейсів від Google AI чи Amazon). Запити до мережі дозволено повторювати необмежену кількість разів і на основі результатів покращувати ошукуючі зразки.

Тоді є можливість обчислити градієнт наближено, для чого достатньо лише результату класифікації $f(x)$ та відповідного вхідного параметра x, як показано в [11]. Якщо обчислимо часткові похідні наближено, за формулою

$$\frac{\partial J(\mathbf{x})}{\partial \mathbf{x}_i} \approx \lim_{h \to 0} \frac{J\left(\mathbf{x} + he_i\right) - J\left(\mathbf{x} - he_i\right)}{2h}, \tag{3}$$

де e_i - вектор, в якому i-тий елемент дорівнює одиниці, а всі решта – нулю, то зможемо застосувати для атаки на чорну скриньку алгоритми, що використовують похідні для розв'язання задачі оптимізації, такі як швидкий градієнтний спуск.

При використанні наближених значень похідних в методі швидкого градієнта (2) вдається досягти результативності, близької до тої, яку отримуємо під час атак на білу скриньку. Це зумовлено тим, що для ефективної роботи методів достатньо лише знаку градієнту. Те, що значення обчислені з певною похибкою, значним чином не впливає на результат. Недоліком такого сценарію є необхідність виконувати класифікацію значну кількість разів, що займає багато часу та ресурсів і може бути легко виявлено.

4.3 Метод Карліні і Вагнера [TODO, optional]

4.4 Перенесення ошукуючих зразків [TODO, optional]

5 Принципи захисту від ошукуючих атак

Як бачимо з попереднього розділу, для атак, які базуються на створенні ошукуючих зразків, характерним є використання градієнтів, обчислених для оцінки чутливості мережі до зміни вхідних даних. Такі градієнти також називають ошукуючими [2]. Очевидно, що при великих значеннях ошукуючих градієнтів зловмиснику значно простіше знайти відповідний ошукуючий зразок для деякого вхідного зображення, бо невеликі збурення вхідного зразка призведуть до значної зміни результату передбачення.

Таким чином, щоб збільшити стійкість мережі до ошукуючих атак, необхідно зменшити таку різку залежність від вхідних даних, а отже і амплітуду коливань ошукуючих градієнтів. Це означає що нам потрібно "згладити" модель, отриману під час стандартного процесу навчання, щоб допомогти їй краще узагальнювати зразки, які не є частиною тренувального набору.

Варто звернути увагу на те, що ошукуючі зразки не обов'язково мають бути створені штучно на основі дослідження мережі-цілі, спеціально для того щоб отримати бажаний результат класифікації. У цьому розділі ми побачимо, що в "природі" таких прикладів також вдосталь, причому деякі неправильно натреновані нейронні мережі є до них особливо вразливими.

В попередніх розділах ми визначили надійність глибоких нейронних мереж як їх стійкість до обмежених збурень вхідних даних. Іншими словами, надійна модель повинна показувати високу точність не лише всередині навчального набору даних, а й за його межами, тобто моделювати функцію, яка "інтуїтивно" розподіляє віхідні дані між запропонованими категоріями в околі даного зразка. Цей окіл може бути визначений нормою у відповідному просторі.

[TODO] [5] Надійність використання норми при визначенні стійкості до атак

5.1 Боротьба з штучно створеним шумом ? природа ошукуючих зразків

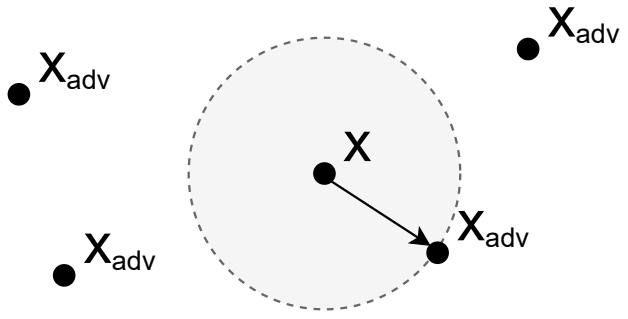

Рисунок 2. 2D ілюстрація.

Не просто знайти ефективний підхід до боротьби з ошукуючими збуреннями, при цьому не завдавши шкоди точність моделі.

Adversarial training

Example detection

Generalization (зменшення розмірності)

Додавання випадкового шуму

5.2 Надмірне тренування як причина вразливості

Працюючи ... На думку спадає ще один феномен у машинному навчанні, який має схожу причину – *перетренування (overfitting)*.

І справді, бачимо залежність між стійкістю моделі та надмірним тренуванням. Перетреновані моделі надзвичайно вразливі до штучно створених ошукуючих зразків. Більше того, нівіть класифікація зразків з тестувального набору є для них складною задачею.

Така поведінка зумовлена тим, що границя рішень .....

6 Захисна дистиляція та її модифікації

В цьому розділі ми розглянемо підхід до оптимізації нейронної мережі або групи нейронних мереж, метою якого є зменшення обчислювальних затрат та ресурсів під час функціонування моделі. Це підхід був розроблений інженерами компанії Google Хілтоном на ін. і описаний в [3]. Не зважаючи на те, що першочерговою метою дистиляції є зменшення розміру нейронної мережі щоб зробити можливим розгортання на пристроях з обмеженими ресурсами, згідно з [2] її побічні ефекти можуть зробити мережу стійкішою до ошукуючих зразків.

6.1 Дистиляція нейронних мереж

Основна ідея методу дистиляції полягає у перенесенні знань з мережі прообразу за допомогою векторів імовірності приналежності зразка до деякого класу, отриманих використовуючи вже існуючу нейронну мережу. На основі цих даних тренується нова нейронна мережа з меншою розмірністю, при чому без значних втрат точності. Ця ідея базується на тому, що знання, здобуті моделлю під час тренування можна отримати не лише безпосередньо з ваг НМ, вони також "закодовані" в її передбаченнях.

Як ми уже згадували раніше, кожен результат передбачення містить у собі деяку інформацію про модель. Таким чином вектори імовірності $\hat{y}$, отримані з одної моделі містять у собі достатньо інформації, щоб з їх допомогою можна було відтворити модель. Дистиляція "витягує" знання з цих векторів і переносить їх на модель з менш комплексною архітектурою.

Використовуючи уже натреновану та готову до використання модель, кожному зразку тренувальних даних ставиться у відповідність вектор ймовірностей приналежності зразка до кожного з класів, які належать до предметної області моделі. Перевага використання таких прогнозів полягає у тому, що тепер кожна мітка містить не просто жорсткі оцінки, а інформацію про приналежність зразка до кожного з класів.

Процедура тренування прообразу не має строгих обмежені і нічим не відрізняється від звичних підходів. Наприклад, можна застосувати градієнтний спуск на даних з тренувального набору. Тепер можемо ввести поняття передавального набору.

Означення 6.1 (TODO) *Означення передавального набору.*

В найпростішій формі дистиляції знання переносяться до дистильованої моделі через тренування на вище згаданому передавальному наборі даних, використовуючи м'які мітки отримані при високих температурах вихідного шару моделі-оригіналу. Така ж температура використовується під час тренування дистильованої моделі, але після завершення процесу її повертають до стандартного значення, рівного одиниці.

[TODO] Якісь припущення?

Проблема цього підходу криється в тому, що точність моделі-прообраза завжди є менша за 100%. Це означає, що передавальний набір буде містити деякі зразки з неправильними мітками і тренування на ньому нової моделі призведе до втрат точності на тестувальному наборі. Якщо відомі точні класи об'єктів, приклади з неправильними мітками варто виключити з передавального набору або замінити їх на правильні жорсткі мітки. Таким чином процес дистиляції може бути значно покращеним і точність моделі-образу залишиться на високому рівні.

Ще один підхід до розв'язання проблем з передавальним набором наводять автори в [3]. Під час тренування вони пропонують використовувати середнє арифметичне двох різних функцій оцінки. Перша з них – функція крос ентропії [TODO] ref, обчислена з використанням такої ж високої температури T в вихідному шарі НМ, яка використовується під час генерації м'яких міток. Друга функція оцінки – функція крос ентропії з правильними жорсткими мітками при температурі $T = 1$.

[TODO] формула

Оскільки величини градієнтів масштабуються з коефіцієнтом $\frac{1}{T^2}$, то варто домножити їх на T^2 при використанні як м'яких, так і жорстких міток. Це гарантує, що відносні частки вкладу м'якої та жорсткої міток залишатимуться приблизно незмінними, якщо температура буде змінюватися під час експериментів.

[TODO] формула

6.2 Процес тренування та основні параметри

Рисунок 3. Схема тренування/ схема НМ.

Процес дистиляції розпочинається з підготовки мережі-прообразу, вихідний шар якої побудований на основі функції – Softmax з врахуванням температури випаровування T.

[TODO] ref Формула

$$\hat{Y}(x) = \left(\frac{e^{z_i(x)/T}}{\sum_{i=0}^{C-1} e^{z_i(X)/T}} \right)_{i \in 0 \ldots C-1} \tag{6.1}$$

У межах Softmax шару кожен нейрон відповідає класу за індексом $i \in 0 \ldots -1$ (де C - кількість класів) і обчислює відповідний компонент за формулою 6.1, де $Z(x) = z_0(X), \ldots, z_{-1}$ – C логітів [TODO](перекласти/ перефразувати/ послатись на розділ про НМ), що відповідають виходу прихованого шару для кожного з C класів у наборі даних.

Параметр T також називають температурою дистиляції. Він T впливає на те, наскільки елементи вектору ймовірностей $\hat{y}$ будуть відрізнятись один від одного.

[TODO] Температура відіграє центральну роль у основних явищах дистиляції, що ми покажемо пізніше у цьому розділі.

Так при малих T різниця між ймовірностями буде значною, тобто один із елементів вектора буде сильно виділятись з-поміж інших. В свою чергу, висока температура змушує НМ створювати ймовірнісні вектори з відносно великими значеннями для кожного класу.

Рисунок 4. Залежність ймовірностей для конкретного класу від температури дистиляції [TODO] bar plots.

Дійсно, при високій температурі, логіти у векторі Z(x) стають незначними у порівнянні з температурою . Тому всі компоненти вектор імовірності $\hat{Y}(X)$, виражені у рівнянні (6.1) прямують до $1/C$ при $T \to \inf$. Чим вища температура, тим більш неоднозначним буде розподіл ймовірностей (тобто всі ймовірності виходу $\hat{Y}(X)$ близькі до $1/C$), тоді як при малій температурі розподіл буде більш дискретним (тобто одна з ймовірностей у виході $\hat{Y}(X)$ буде близькою до 1, а решта – близькі до 0).

На основі векторів ймовірності, створених першою НМ, формується передавальний набір

6.3 Оцінка захисту

Фільтрований передавальний набір vs не фільтрований

7 Методи з гарантованою стійкістю

Основна проблема багатьох методів захисту, зокрема і розглянутої раніше захисної дистиляції полягає в тому, що їх ефективність важко, або навіть неможливо довести теоретично. Через це часто виникають випадки, коли захист моделі є ефективним лише для конкретних задач і проти конкретних методів атаки, як, наприклад, в [TODO] cite distilation is not robust. Тому цінними є методи, при використанні яких можна гарантувати певний рівень стійкості. Такі методи захисту також називають *сертифікованими*. Один з таких методів ми і роглянемо в даному розділі. Він називається *PixelDP* і був запропонований запропонований Матіасом Лекуєром та ін. в [4]. Він базується на принципі *диференційованої конфіденційності (differential privacy)*, в основі якого лежить рандомізація обчислень на великих масивах даних [7].

[TODO]

7.1 Захист PixelDP

Рисунок 5. Схема тренування/ схема НМ.

7.2 Оцінка захисту

8 Комбінація методів та побудова архітектури, стійкої до атак

9 Експерименти?

Висновок

Додатки

Додаток 1. Деякі приклади ошукуючих зразків

Додаток 2. Таблиці про нм?

Додаток 3. Нм для розпізнавання цифр [TODO](prev work)

Додаток

Література

[1] *Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, Rob Fergus* / Intriguing properties of neural networks / arXiv preprint arXiv:1312.6199 (2014)

[2] *Nicolas Papernot, Patrick McDaniel, Xi Wu Somesh Jha,Ananthram Swami* / Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks / arXiv preprint arXiv:1511.04508 (2016)

[3] *Geoffrey Hinton, Oriol Vinyals, Jeff Dean* / Distilling the Knowledge in a Neural Network / arXiv preprint arXiv:1503.02531 (2015)

[4] *Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, Suman Jana* / Certified Robustness to Adversarial Examples with Differential Privacy/ arXiv preprint arXiv:1802.03471 (2019)

[5] *Alhussein Fawzi, Omar Fawzi, Pascal Frossard* / Analysis of classifiers' robustness to adversarial perturbations / arXiv preprint arXiv:1502.02590 (2016)

[6] *Thamas G. Dietterich* / Ensemble Methods in Machine Learning / Springer (2000)

[7] *Cynthia Dwork* / Differential Privacy / Automata, Languages and Programming. ICALP (2006)

[8] *Ian Goodfellow, Jonathon Shlens, Christian Szegedy* / Explaining and Harnessing Adversarial Examples / arXiv preprint arXiv:1412.6572 (2014)

[9] *Alfio Quarteroni, Riccardo Sacco, Fausto Saleri* / Numerical Mathematics / – Springer, 2000. –300 p.

[10] *Nicolas Papernot, Patrick McDaniel, Ian Goodfellow* / Transferability in machine learning: from phenomena to black-box attacks using adversarial samples / arXiv preprint arXiv:1605.07277 (2016)

[11] *Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, Cho-Jui Hsieh* / ZOO: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks without Training Substitute Models / arXiv preprint arXiv:1708.03999 (2017)

[12] *Nicholas Carlini, David Wagner* / Towards Evaluating the Robustness of Neural Networks / arXiv preprint arXiv:1608.04644 (2017)

[13] *Богдан Бугрій, Юрій Музичук* / Атаки на глибокі нейронні мережі / (2020)