Методы слияния данных для эффективного выбора методов

Надафф М.Н.

Российский университет дружбы народов им. П. Лумумбы

Примечание автора

Первый пункт: Объяснение данных и предобработки данных

Второй пункт: Архитектура модели и техника слияния данных

Третий пункт: Результаты и отслеживание модели

Методы слияния данных для эффективного выбора методов

Целью данного эксперимента было установление эталона для будущих тестирований и сравнений методов слияния данных. Этот фундаментальный этап позволяет проводить стандартизированную оценку различных методов, способствуя значимым сравнениям и выявлению наиболее эффективных подходов.

Первый пункт: **Объяснение данных и предобработки данных**

Первоначальной задачей данного исследования стало нахождение надёжного, хорошо документированного и общедоступного многовидового датасета, пригодного для изучения методов слияния данных. После длительного поиска был найден соответствующий набор данных, опубликованный одной из московских больниц, содержащий КТ-сканы грудной клетки в формате NIfTI (.nii), используемые для диагностики COVID-19.

Каждое объемное изображение охватывает всю область лёгких, включая 5 см выше и 5 см ниже их границ, что обеспечивает полное 3D-представление анатомии грудной клетки. Данные классифицированы по степени выраженности симптомов от **CT-0** (отсутствие признаков COVID-19) до **CT-4** (тяжёлое течение заболевания). С увеличением номера класса увеличивается и степень поражения.

Каждый том состоит из переменного количества 2D-срезов, каждый размером **512×512 пикселей**. Ввиду различного количества срезов между сканами была проведена

предварительная обработка данных. На первом этапе были нормализованы значения единиц Хаунсфилда (HU) в диапазон [0, 1]. HU — это шкала, используемая для количественного описания рентгеновской плотности различных тканей, где воздух, мягкие ткани и кости имеют разные значения. Нормализация снижает разброс данных и улучшает сходимость нейросетей при обучении.

Затем тома были приведены к фиксированной глубине **50 срезов** с помощью линейной интерполяции — это обеспечило единообразие по глубине. Размер каждого среза был уменьшен до **224×224 пикселей**, что соответствует стандартным входным параметрам для сверточных нейронных сетей.

Для упрощения задачи классификации все классы были объединены в бинарный формат: **CT-0** остался как класс «без COVID», а **CT-1 до CT-4** были объединены в единый класс **CT-23**, обозначающий наличие COVID-19. Таким образом, задача фокусируется на выявлении наличия заболевания, а не на его степени тяжести.

Каждый файл .nii имел уникальный идентификатор пациента. С целью имитации многовидового представления каждый том был программно разделён на две равные части, которые сохранялись как split_part_1.nii и split_part_2.nii в директории, соответствующей ID пациента. Например:

Processed_CT/study_0001/split_part_1.nii

Processed_CT/study_0001/split_part_2.nii

Такая стратегия позволяет моделировать ситуацию, когда разные углы обзора объекта предоставляют дополняющую друг друга информацию — аналогично многовидовому изображению 3D-объекта. Этот подход стал основой для последующего применения методов слияния данных в архитектуре модели.

После завершения всех этапов обработки, итоговый набор данных — нормализованный, стандартизированный и адаптированный под многовидовое обучение — получил название **Processed_CT** и был загружен на платформу **Kaggle.com** для дальнейшей разработки и обучения моделей.

Второй пункт: Архитектура модели и техника слияния данных

После завершения этапов предобработки и балансировки классов, итоговый набор данных сократился до 172 примеров на каждый класс (то есть 172 случая без COVID-19 и 172 случая с подтверждённым COVID-19). Такое количество данных считается крайне малым для задач глубинного обучения, однако это было обусловлено как преднамеренными, так и внешними ограничениями.

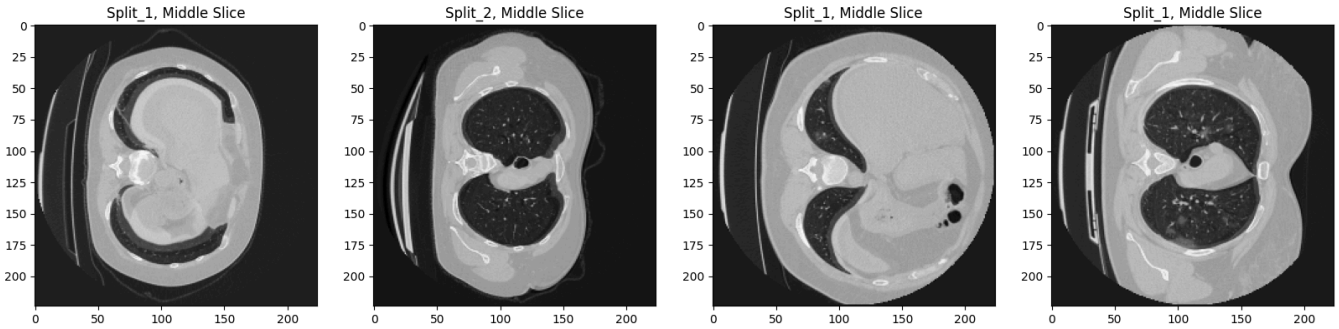
С одной стороны, намеренное сокращение объёма данных было связано с ограниченными аппаратными ресурсами — в частности, с недостаточной доступностью видеокарт (GPU), что ограничивало допустимую сложность и размер модели. С другой стороны, исходный набор данных сам по себе был небольшим, сильно несбалансированным, и содержал множество некорректно размеченных или неподтверждённых образцов, которые пришлось исключить в ходе ручной фильтрации и валидации.

Для эффективной работы с таким небольшим объёмом данных была реализована пользовательская система подачи данных (data pipeline), основанная на библиотеке `Sequence` из Keras. Это позволило организовать пакетную (batch-wise) и поэтапную подачу данных напрямую в модель, обеспечивая эффективную работу с памятью и стабильное обучение.

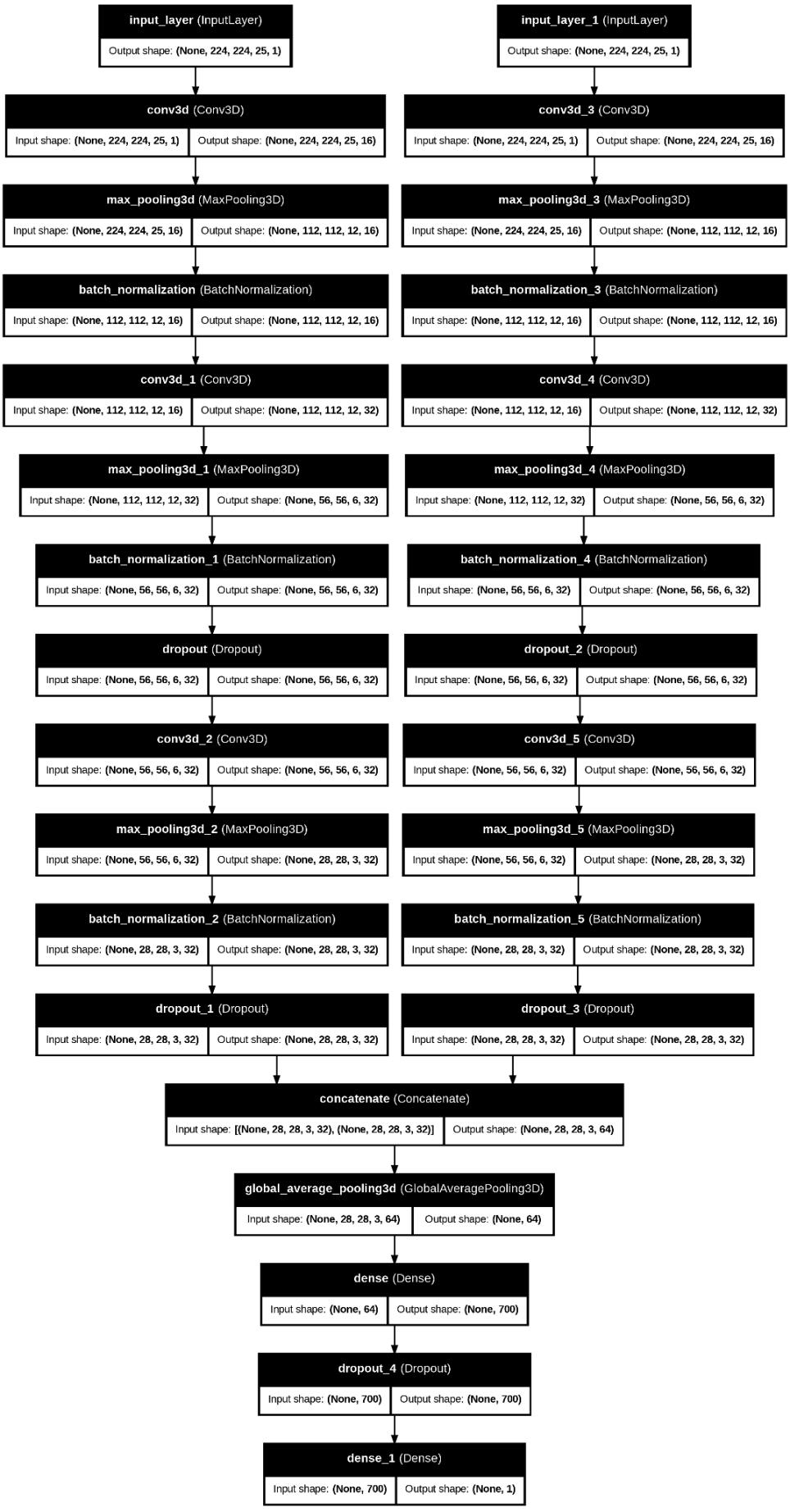
Перед подачей данных в модель все .nii-объёмы были преобразованы в массивы NumPy. Затем, с помощью функции `train_test_split` из библиотеки scikit-learn, данные были разделены на обучающую и тестовую выборки. Для обучения использовался размер пакета (batch size) равный 4, а для тестирования — 2, что позволило снизить риск переобучения и соблюсти ограничения по памяти.

Особое внимание было уделено аугментации данных в обучающей выборке, так как разнообразие входных данных критично при небольшом наборе. В рамках аугментации

объёмы КТ вращались на заранее заданные углы, имитируя разные положения тела пациента и различные ракурсы сканирования. Полученные объёмы помещались в словарь NumPy-массивов и использовались в дальнейшем обучении модели.



После настройки пайплайна и подготовки данных начались эксперименты с архитектурой модели. Путём многочисленных попыток и ошибок была выбрана архитектура 3D сверточной нейронной сети (3D CNN), показавшая наилучшие результаты по метрикам точности и полноты.


input_layer (InputLayer)
Output shape: (None, 224, 224, 25, 1)
input_layer_1 (InputLayer)
Output shape: (None, 224, 224, 25, 1)
conv3d (Conv3D)
Input shape: (None, 224, 224, 25, 1)
Output shape: (None, 224, 224, 25, 16)
conv3d_3 (Conv3D)
Input shape: (None, 224, 224, 25, 1)
Output shape: (None, 224, 224, 25, 16)
max_pooling3d (MaxPooling3D)
Input shape: (None, 224, 224, 25, 16)
Output shape: (None, 112, 112, 12, 16)
max_pooling3d_3 (MaxPooling3D)
Input shape: (None, 224, 224, 25, 16)
Output shape: (None, 112, 112, 12, 16)
batch_normalization (BatchNormalization)
Input shape: (None, 112, 112, 12, 16)
Output shape: (None, 112, 112, 12, 16)
batch_normalization_3 (BatchNormalization)
Input shape: (None, 112, 112, 12, 16)
Output shape: (None, 112, 112, 12, 16)
conv3d_1 (Conv3D)
Input shape: (None, 112, 112, 12, 16)
Output shape: (None, 112, 112, 12, 32)
conv3d_4 (Conv3D)
Input shape: (None, 112, 112, 12, 16)
Output shape: (None, 112, 112, 12, 32)
max_pooling3d_1 (MaxPooling3D)
Input shape: (None, 112, 112, 12, 32)
Output shape: (None, 56, 56, 6, 32)
max_pooling3d_4 (MaxPooling3D)
Input shape: (None, 112, 112, 12, 32)
Output shape: (None, 56, 56, 6, 32)
batch_normalization_1 (BatchNormalization)
Input shape: (None, 56, 56, 6, 32)
Output shape: (None, 56, 56, 6, 32)
batch_normalization_4 (BatchNormalization)
Input shape: (None, 56, 56, 6, 32)
Output shape: (None, 56, 56, 6, 32)
dropout (Dropout)
Input shape: (None, 56, 56, 6, 32)
Output shape: (None, 56, 56, 6, 32)
dropout_2 (Dropout)
Input shape: (None, 56, 56, 6, 32)
Output shape: (None, 56, 56, 6, 32)
conv3d_2 (Conv3D)
Input shape: (None, 56, 56, 6, 32)
Output shape: (None, 56, 56, 6, 32)
conv3d_5 (Conv3D)
Input shape: (None, 56, 56, 6, 32)
Output shape: (None, 56, 56, 6, 32)
max_pooling3d_2 (MaxPooling3D)
Input shape: (None, 56, 56, 6, 32)
Output shape: (None, 28, 28, 3, 32)
max_pooling3d_5 (MaxPooling3D)
Input shape: (None, 56, 56, 6, 32)
Output shape: (None, 28, 28, 3, 32)
batch_normalization_2 (BatchNormalization)
Input shape: (None, 28, 28, 3, 32)
Output shape: (None, 28, 28, 3, 32)
batch_normalization_5 (BatchNormalization)
Input shape: (None, 28, 28, 3, 32)
Output shape: (None, 28, 28, 3, 32)
dropout_1 (Dropout)
Input shape: (None, 28, 28, 3, 32)
Output shape: (None, 28, 28, 3, 32)
dropout_3 (Dropout)
Input shape: (None, 28, 28, 3, 32)
Output shape: (None, 28, 28, 3, 32)
concatenate (Concatenate)
Input shape: [(None, 28, 28, 3, 32), (None, 28, 28, 3, 32)]
Output shape: (None, 28, 28, 3, 64)
global_average_pooling3d (GlobalAveragePooling3D)
Input shape: (None, 28, 28, 3, 64)
Output shape: (None, 64)
dense (Dense)
Input shape: (None, 64)
Output shape: (None, 700)
dropout_4 (Dropout)
Input shape: (None, 700)
Output shape: (None, 700)
dense_1 (Dense)
Input shape: (None, 700)
Output shape: (None, 1)

Окончательная архитектура предполагала два входных потока — по одному на каждую из двух частей ранее разделённого объёма КТ. Каждый поток обрабатывался идентичным модулем 3D CNN, после чего признаки, извлечённые из обоих входов, объединялись через операцию конкатенации. Затем данные проходили через слой глобального среднего объединения (Global Average Pooling), полносвязный слой (Dense) и, наконец, попадали на выходной слой, который классифицировал объём как положительный или отрицательный по признакам COVID-19.

Такая архитектура была разработана с целью имитации мульти ракурсного анализа, аналогичного тому, как врач-рентгенолог оценивает несколько сечений и ракурсов КТ перед постановкой диагноза. Модель обучалась извлекать комплементарную информацию из различных “видов” одного и того же анатомического объекта.

Параметры итоговой модели:

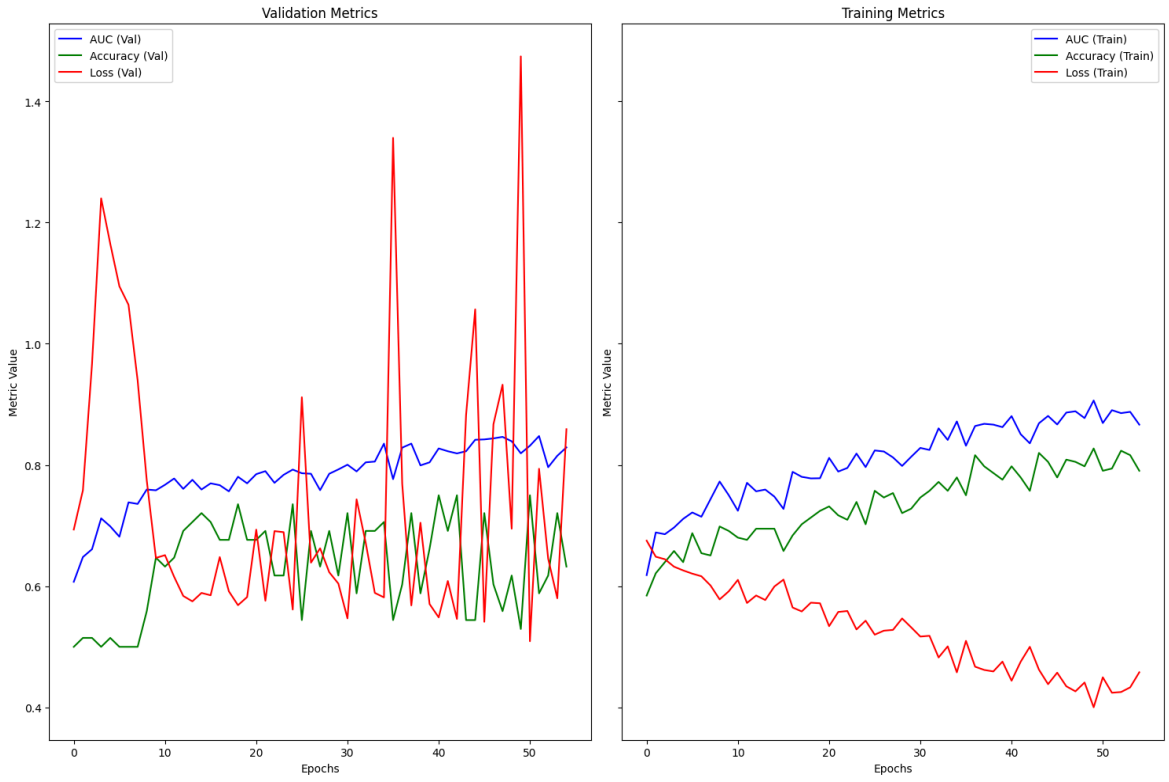
- **Общее число параметров:** 130,809 (510.97 КБ)
- **Обучаемых параметров:** 130,489 (509.72 КБ)
- **Необучаемых параметров:** 320 (1.25 КБ)

Для мониторинга обучения использовались различные callbacks, включая отслеживание использования видеопамяти (GPU memory). Этот трекер в реальном времени показывал, сколько памяти (в мегабайтах) потребляется в каждом цикле обучения, что позволило адаптировать модель под доступные аппаратные ресурсы.

Метрики точности, результаты обучения и сравнительный анализ представлены в следующем разделе.

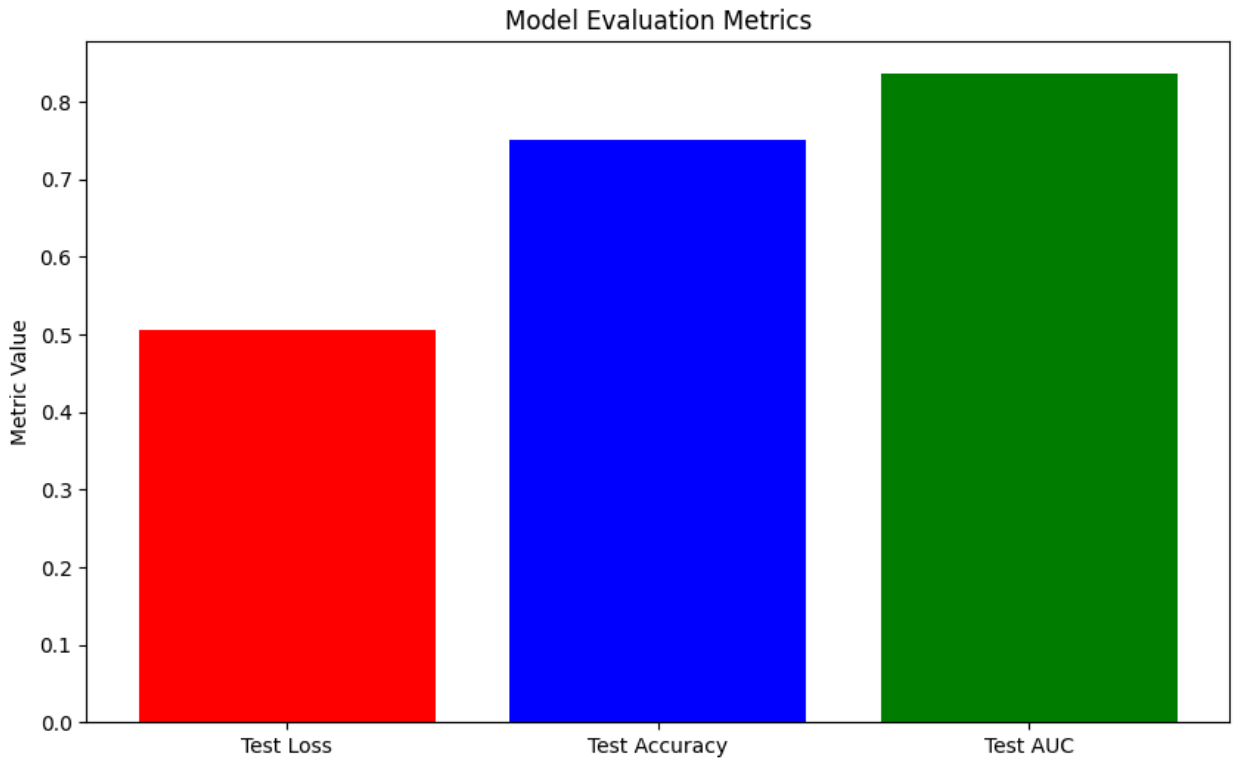
Третий пункт: Результаты и отслеживание модели

После завершения этапа обучения модели, были получены следующие результаты.



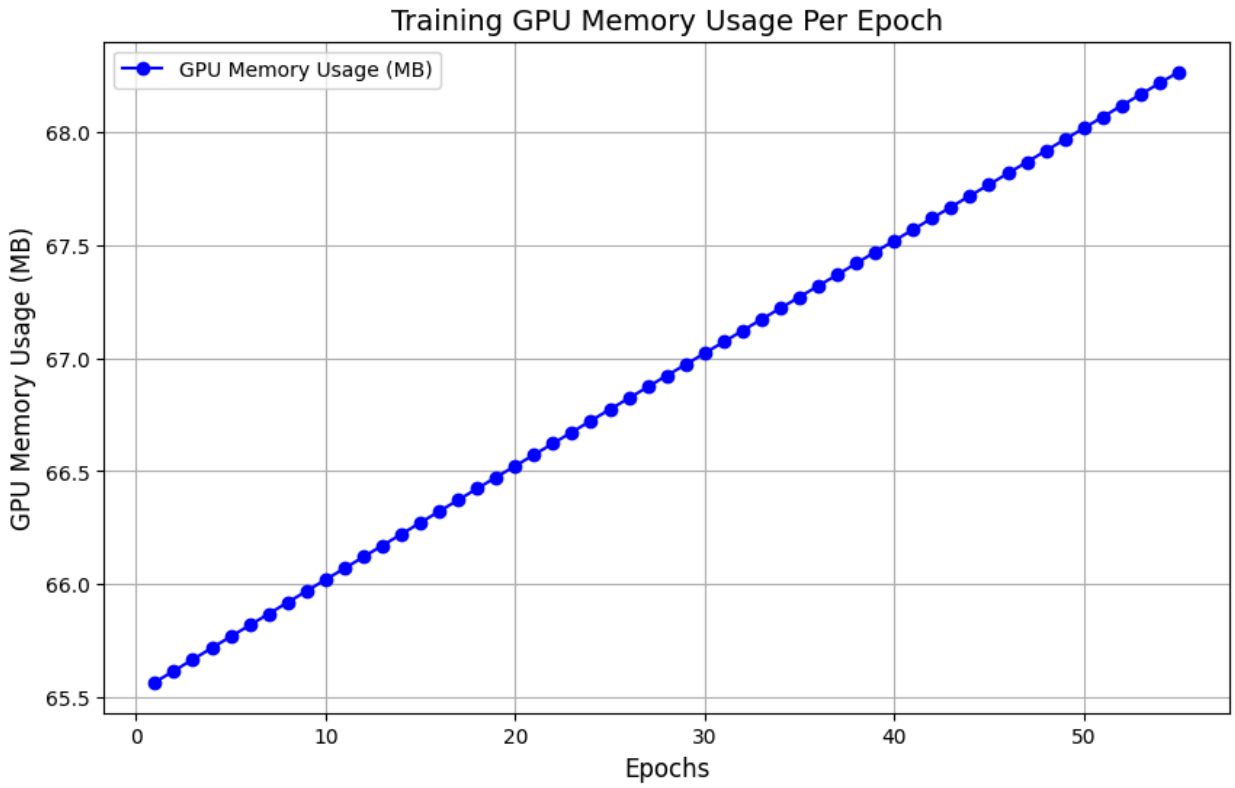
Во время обучения точность (accuracy) и площадь под кривой (AUC) для обучающего набора данных демонстрировали относительно плавный рост, что говорит о том, что модель последовательно адаптировалась к тренировочным данным. Однако, метрики на

валидационной выборке оказались весьма нестабильными — наблюдались резкие колебания и отсутствие чёткой тенденции к улучшению. Такой результат, вероятнее всего, обусловлен крайне малым объёмом валидационного подмножества, что снижает его репрезентативность и увеличивает чувствительность к шуму и случайным выбросам. Это типичная проблема при работе с ограниченными медицинскими наборами данных, где даже пара «нехарактерных» сканов может существенно исказить общую метрику.



Тем не менее, при финальном тестировании модели на отложенной тестовой выборке она показала удовлетворительные результаты, особенно с учётом ограниченного объёма данных, использованных на всех этапах. Значение функции потерь составило **0.5**, точность — **73%**, а AUC достиг **85%**. Последний показатель особенно важен в задачах бинарной классификации, так как он демонстрирует способность модели отличать положительные

классы от отрицательных вне зависимости от порогового значения. Таким образом, можно утверждать, что предложенная архитектура способна извлекать значимые признаки из томографических данных даже при крайне скудной обучающей базе.



Помимо оценки точности и ошибок классификации, в процессе разработки большое внимание уделялось эффективности и стабильности обучения. До начала обучения, при подготовке данных и загрузке их через индивидуальный data pipeline, объём используемой видеопамяти составлял **65.5 МБ**. В ходе обучения этот объём линейно увеличился до **около 69 МБ**. Такой плавный рост без резких скачков или падений указывает на стабильную загрузку данных и отсутствие «узких мест» (bottlenecks) или утечек памяти. Это говорит о том, что pipeline работал корректно, без перегрузок или чрезмерной

аллокации ресурсов, что особенно важно при работе с ограниченными вычислительными мощностями. Время обучения составило **1 час 37 минут**, что также является показателем эффективного использования доступных вычислительных ресурсов.аллокации ресурсов, что особенно важно при работе с ограниченными вычислительными мощностями.

Полный код доступен здесь: https://github.com/MaximNadd/Research

Список литературы

МосМедДата: набор данных из 1110 грудных КТ-сканов, выполненных в период эпидемии COVID-19,

Авторы: Морозов С.П., Андрейченко А.Е., Блохин И.А., Гележ П.Б., Гончар А.П., Николаев А.Е., Павлов Н.А., Чернина В.Ю., Гомболевский В.А.

Организации: Научно-практический клинический центр диагностики и телемедицинских технологий, Департамент здравоохранения города Москвы.

https://jdigitaldiagnostics.com/DD/article/view/46826