АГЛЯД ГАТОВЫХ ПРАГРАМНЫХ РАШЭННЯЎ ПА РАСПАЗНАВАННЮ ЖЭСТАЎ РУК. MEDIAPIPE.

Бекетаў Я.Д., студэнт гр.150504

Беларускі дзяржаўны ўніверсітэт інфарматыкі і радыёэлектронікі
г. Мінск, Рэспубліка Беларусь

Перцаў Д.Ю. – канд. тэхн. навук

У дадзенай працы праведзены агляд гатовага праграмнага рашэння Mideapipe на прыкладзе распазнавання жэстаў рук.

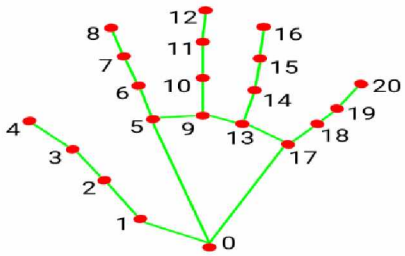
Уводзіны. У апошняе 10-годдзе вельмі актыўна і з адпаведнымі поспехамі аднавіла свае развіццё сфера штучнага інтэлекту. З-за сваёй ўніверсальнасці, штучны інтэлект усё больш імкнуцца інтэграваць у жыццё чалавека, каб задавальніць яго патрабаванні. Чалавек можа выражаць свае жаданні па-рознаму, але ў аснове гэтага выражэння ляжаць: здольнасць гаварыць і жэстыкуляваць. Жэстыкуляцыя звычайна немагчыма без выкарыстоўвання рук. Каб зразумець жэсты, трэба ведаць іх значэнне, то бок іх эквівалент у мове жэстаў. Але выканаўшы задачу распазнавання, можна атрымаць вельмі шмат важнай інфармацыі. Для рашэння гэтай задачы ёсць сэнс скарыстацца новымі тэхналогіямі ў сферы ШІ. Такой тэхналогіяй з'яўляецца, распрацаваны кампаніяй Google фрэймворк з адкрытым зыходным кодам для машыннага навучання пад назвай MediaPipe.

Агляд MediaPipe. Як сведчыць інфармацыя з афіцыйнага сайту [1], гэты фрэймворк здольны прапанаваць лёгкую ў выкарыстоўванні, інавацыйную і сапраўды хуткую ў распрацоўцы і выкананні платформу для машынага навучання ў вялікім шэрагу вобласцяў: відэаназіранні, абпрацоўцы тэксту, гуку, выяўленні аб'ектаў, класіфікацыі відарысаў і распазнаванні жэстаў рук. У дадзеным аглядзе падрабязна разбярэцца апошняе рашэнне, з прывядзеннем прыкладаў рэалізацыі для камп'ютараў.

Рашэнне па распазнаванню жэстаў рук (Hand gesture recognition, HCR), дазваляе як аналізаваць фатаздымкі і відэа, так і выдаваць рэзультат у рэальным часе. Для распазнавання выкарыстоўваецца мадэль машыннага навучання, пры дапамозе каторай вызначаюцца арыентыры (landmark) рук, вызначаецца правая ці левая рука прадстаўлена на відарысу і катэгорыя жэстаў для некалькіх рук адначасова. З-за напрамку на рынак мабільных прыкладанняў, гэтае рашэнне адначасова існуе і для Android (kotlin), IOs (swift, objectiveC), Web (js), Raspberry PI (python) і для Desktop (python). У дадзенным аглядзе будзе прыводзіцца прыклад для камп'ютараў на мове праграмавання Python.

Асаблівасці рашэння HCR. Складанасць вызначэння рукі на фатаздымку абумоўлена непастаяннасцю яе формы. Мадэль, каторую выкарыстоўвае HCR, у першую чаргу скіравана на выяўленне далоні, а ўжо потым на пальцы. Пры гэтым асноўную задачу выконваюць перадзапакаваныя пакеты мадэляў для вызначэння арыентыраў і класіфікацыі жэстаў. У сукупнасці гэтае рашэнне здольна вызначыць 8 класаў: сціснуты кулак, раскрытую далонь, указальны палец, вялікі палец уніз, вялікі палец уверх, знак перамогі ("V"), "I love you" і нявызначаны жэст [2].

Класіфікатар працуе напрамую не з відарысам, а з арыентырамі. У архітэктуру ўваходзіць двухкрокавы нейрасеткавы канвеер з мадэллю устройвання (embedding model), за катарой ідзе мадэль класіфікацыі (classification model). У якасці ўваходу прымаюцца: 21 трохмерныя экранныя арыентыры, што нарміраваныя па памеру відарыса, у выглядзе тэнзару 1 на 63; скаляр з нефіксаванай коскай, каторы адлюстроўвае імавернасць таго, левая ці правая рука на відарысе; 21 трохмерныя экранныя арыентыры, што нарміраваныя адносна зямнога маштабу, у выглядзе тэнзару 1 на 63. На выхад адпраўляецца 8-элементны вектар, у каторым змяшчаецца імавернасць з'яўлення таго, ці іншага класу [3]. У працэсе кожная рука раздзяляецца на рад арыентыраў (гл. малюнак 1).



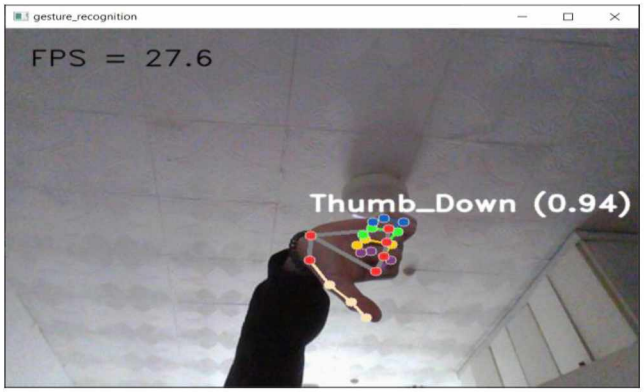
0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Малюнак 1 – Арыентыры рукі

Праграмнае рашэнне для распазнавання жэстаў у рэальным часе. У якасці параўнання было вырашанна напісаць дзве праграмы: адна, з поўным выкарыстоўваннем mediapipe, другая, з выкарыстоўваннем сваёй мадэлі для аналізу арыентыраў. Для двух варыянтаў праграм, абавязковым было выкарыстоўванне бібліятэкі OpenCV.

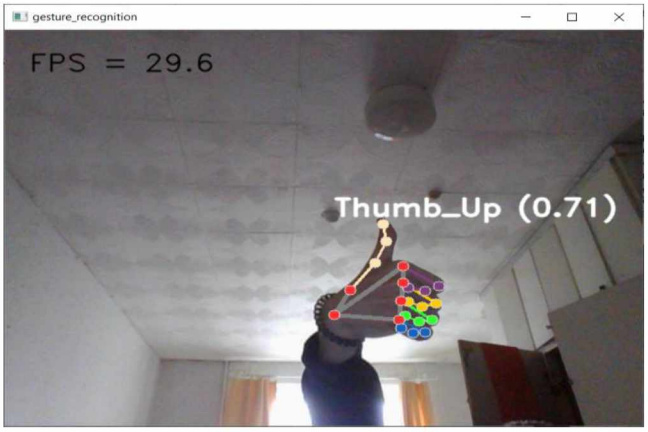
Праграма з поўным выкарыстоўваннем MediaPipe. Адна з асноўных задач у стварэнні сваёй нейроннай сеткі, гэта пошук data set (набора дадзенных), па каторых будзе трэніравацца і правярацца

нейрасетка. Што датычыцца HCR, ужо існуе набор з 30 тыс. фатаздымкаў рук і жэстаў, таму гэтае праблема не ставіцца на разгляд. Вядома, што гатовы набор змяшчае толькі тыя жэсты, каторыя былі апісаны вышэй для гэтага рашэння, аднак MediaPipe дазвялае гнуткасць у гэтым пытанні. Пры наяўнасці набору (дастаткова па 100 фатаздымкаў на жэст) можна стварыць сваю мадэль на аснове той, што выкарыстоўваецца ў фрэймворку. Для гэтага агляду падобны шлях не спатрэбіўся, таму выкарыстоўваўся стандартны набор. На малюнку 2 прадстаўлены вынікі выяўлення жэста «палец уніз» з імваернасцю 94%.



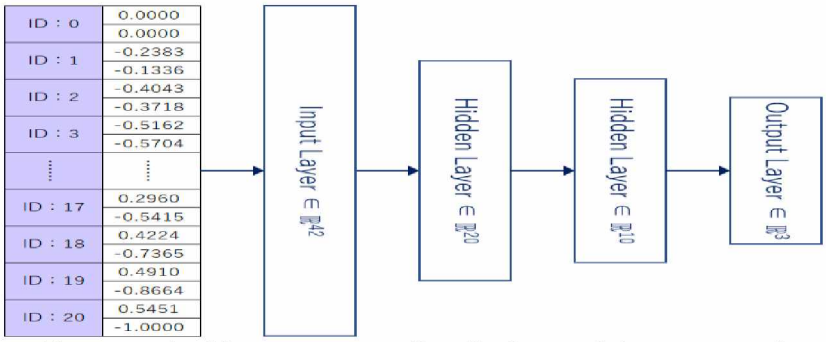
Малюнак 2 – Вынік выяўлення жэста «палец уніз»

Для гэтай праграмы вынікі праверкі выяўлення жэста «палец уверх» паказаны на малюнку 3.



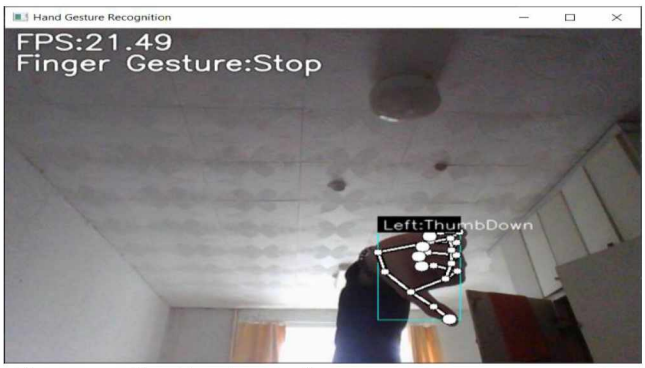
Малюнак 3 – Вынік выяўлення жэста «палец уверх»

Праграма з асабістай абпрацоўкай арыентыраў. У аснове выкарыстоўваецца мадэль MediaPipe, каторая дастае з фатаздымка арыентыры. Далей гэтыя арыентыры нармалізуюцца і адпраўляюцца ў 4-слаёвую мадэль, створанную пры дапамозе TensorFlow [4]. Падрабязна азнаёміцца з адлюстраваннем мадэлі можна на малюнку 4.



Малюнак 4 – Мадэль для асабістай абпрацоўкі арыентыраў

У мадэлі выкарыстоўваюцца такія функцыя актывацыі як ReLu і Softmax. Дадзеная нейрасетка абпрацоўвае csv файл з значэннямі нармалізаваных арыентыраў, і на аснове супадзення з класам, на выхадзе вяртае вектар з імавернасцямі з'яўлення таго, ці іншага класу. На малюнку 5 прадстаўлены вынікі выяўлення жэста «палец уніз».



Малюнак 5 – Вынік выяўлення жэста «палец уніз»

На малюнку 6 прадстаўлены вынікі выяўлення жэста «палец уверх».

Малюнак 6 – Вынік выяўлення жэста «палец уверх»

Параўнанне. Першыя недахопы праяўляюцца ў характары падлікаў, каторыя выкарыстоўваюцца ў другой праграме. Калі браць гатовае рашэнне ад MediaPipe цалкам, вялікая колькасць падлікаў выконваецца пры дапамозе так званых аптымізаваных калькулятараў. Пры пападанні ў камеру рукі і пры яе аналізе, у другой праграмы сярэдні значэнне фпс - 20. Для першай – 30 фпс. Першая праграма стартуе за 7 сек, другая за 14 сек.

Каб паменшыць уплыў недахопу набору дадзеных для другой праграмы, быў зроблены рэжым збору арыентыраў для вызначага класу пад час выканання. Гэта значна палягчае працэс набыцця дадзеных, бо ў выпадку з MediaPipe трэба выкарыстоўваць рэальныя фатаздымкі, што патрабуе больш часу і большай выбаркі. Пры выкарыстоўванні CPU (11th Gen Intel(R) Core(TM) i5-1135G7 2.40GHz 2.42 GHz), пажадана быць падключанным да падзарадкі. Калі не, тады час адказу першай праграмы значна павялічваецца, а ў другой сярэдняе значэнне фпс зніжаецца да 15.

Пры слушным карыстанні MediaPipe можна дабіцца лепшых рэзультатаў у распазнаванні рук, але адначасова, трэба мець дастаткова вялікі набор дадзенных. Працэс распрацоўкі самога прыкладання пры дапамозе MediaPipe сапраўды вельмі хуткі, таму не з'яўляецца праблемай.

Спіс выкарыстаных крыніц:

1. MediaPipe [Электронны рэсурс] – Рэжым доступу: https://developers.google.com/mediapipe. – Дата доступу: 05.04.2024

2. hand-gesture-recognition-mediapipe [Электронны рэсурс] – Рэжым доступу: https://github.com/kinivi/hand-gesture-recognition-mediapipe

3. MediaPipe Hand Gesture Classification [Электронны рэсурс] – Рэжым доступу: https://storage.googleapis.com/mediapipeassets/gesture_recognizer/model_card_hand_gesture_classification_with_faireness_2022.pdf. – Дата доступу: 05.04.2024.

4. Hand Gesture Recognition by Hand Landmark Classification [Электронны рэсурс] – Рэжым доступу: https://www.jstage.jst.go.jp/article/isase/ISASE2022/0/ISASE2022_1_34/_pdf/-char/en. – Дата доступу: 05.04.2024.

АНАЛИЗ ВЛИЯНИЯ РАЗНЫХ ПОДХОДОВ ВЗАИМОДЕЙСТВИЯ С ДАННЫМИ НА ПРОЦЕСС И РЕЗУЛЬТАТ МАШИННОГО ОБУЧЕНИЯ

Крачковский А.В. студент группы 050502

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Перцев Д.Ю. – доцент кафедры ЭВМ, кандидат технических наук

Было проведено исследование, направленное на оценку влияния различных подходов взаимодействия с данными на процесс обучения модели машинного обучения.

Есть много статей [1,2,3], которые рассказывают о том, как подготовить данные для обучения модели, но при этом не удаётся найти, каким именно образом нужно загружать уже подготовленные данные в модель. Цель данного исследования заключалась в анализе процесса обучения модели, способной определять эмоции людей на основе фотографий или видеопотока, с использованием различных способов взаимодействия с набором данных.

Прежде чем начать анализ нужно установить входные данные. Для машинного обучения используется фреймворк PyTorch. Входной слой состоит из 48x48 нейронов, каждый из которых соответствует пикселю подающейся картинки. В скрытой области находится 4 слоя с 2048 нейронами в каждом. Количество нейронов в слое подобрано эмпирическим путём. Выходной слой состоит из семи нейронов, каждый из которых отвечает за одну из эмоций. Список обучаемых эмоций включает в себя злость, отвращение, ужас, радость, спокойствие, грусть и удивление. Набор данных состоит из заранее подготовленных изображений размером 48x48 пикселей. На изображениях находятся люди, которые испытывают эмоцию, которая заранее указана в названии файла изображения.

Данная модель не является конечным вариантом, поэтому результат классификации может быть ошибочным для человека, но не для самой модели. Из этого вытекает следующий вопрос: как оценить влияние данных на обучение? В это случае принято решение использовать среднее значение коэффициента ошибки по функции «Adam» [4]. Модель, используя ранее упомянутый набор данных, будет пытаться определить все эмоции людей по очереди. После анализа всех изображений, модель соберет все коэффициенты ошибок для каждой эмоции и посчитает их среднее значение. Таким образом, можно будет отследить процесс обучения модели и влияние данных в целом.

Первый подход — это использование полного набора данных без изменений. В этом случае хватило тринадцати эпох, чтобы выровнять график функции ошибки (см. рисунок 1).

Рисунок 1 — Результат обучения модели с использование набора данных без изменений.

Можно заметить резкий скачок в начале графика, вызванный неравномерностью набора данных, из-за того, что данных для обучения эмоции «отвращение» меньше, чем остальных, среднее значение ошибки в нем становится больше, чем у остальных. Также можно заметить «ступенчатый» вид графика после скачка. Можно выделить две причины появления такой структуры: модель забывает старые

данные или модель ожидает «увидеть» изучаемую эмоцию из-за прихода новых данных. Небольшой подъём на конце обусловлен неравномерностью данных.

Результаты можно попытаться улучшить, перетасовав набор данных (см. рисунок 2). Проблема, связанная с неравномерностью данных, не исчезает, но график ошибок стремится к выравниванию, что означает более точный результат по сравнению с предыдущим способом обучения. Тем не менее расхождение коэффициента ошибки требует отдельного внимания и следующие способы обучения будут настроены на то, чтобы уменьшить расхождение этого коэффициента.

Рисунок 2 — Результат обучения модели с перемешанным набором данных.

В этом случае мы можем подсчитать наибольшую ошибку и начать работать относительно эмоции с наибольшим коэффициентом ошибки. Использование данного подхода может привести к увеличению времени, затрачиваемого на тестирование модели, что в итоге приведёт к замедлению обучения самой модели. В результате обучения (см. рисунок 3) можно заметить необычную ситуацию — резкие перепады ошибки. После анализа было выявлено, что модель начинает «хитрить». Наименьший коэффициент ошибки имеет эмоция, которая подвергалась обучению в последний момент. Возникла теория, согласно которой данное явление может быть обусловлено необходимостью настройки весов связей между нейронами и, поскольку для обучения используется только одна эмоция, все веса меняются в пользу определения одной конкретной эмоции.

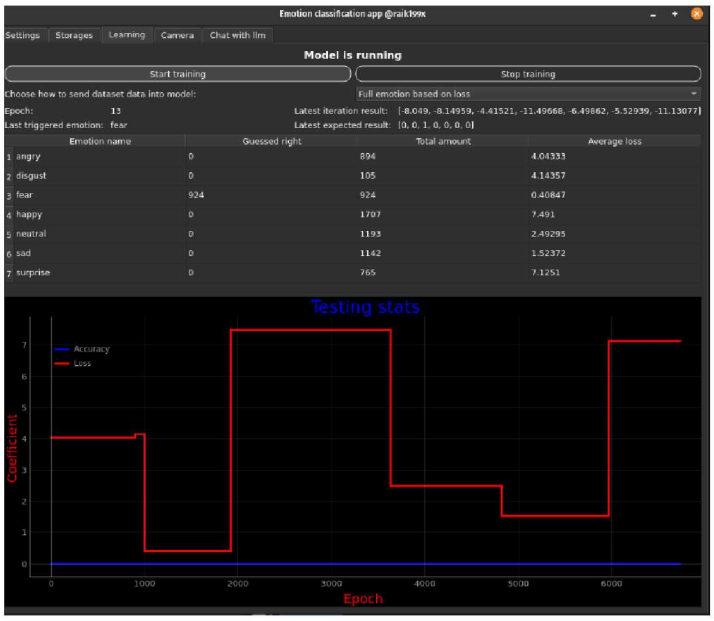


Рисунок 3 — Результат обучения модели при учете наибольшей ошибки и обработки одной конкретной эмоции целиком.

Последним рассматриваемым вариантом является отправка одного изображения эмоции относительно наибольшего коэффициента ошибки. В этом случае проблема неравномерности данных в наборе исчезает, но само обучение сильно замедляется, так как большая часть времени уходит на тестирование, а не на обучение. Как результат, можно интерактивно наблюдать за обучением, но итогового результата ждать придётся долго, что изображено на рисунке 4. Коэффициент ошибки выравнивается у всех эмоций, что имеет очень хороший эффект –, все эмоции имеют одинаковый шанс на определение.

Рисунок 4 — Результат обучения модели при использовании одной фотографии с учётом коэффициента ошибки.

Вывод можно сделать следующий: способ взаимодействия с данными влияет на дальнейшее обучение, но нет одного эталонного способа, который можно было бы использовать во всех случаях, однако можно комбинировать рассмотренные в статье способы взаимодействия с набором данных для улучшения результатов обучения.

Список использованных источников:

1. Разметка данных в машинном обучении: процесс, разновидности и рекомендации [Электронный ресурс]. - Режим доступа: https://habr.com/ru/articles/678524/ - Дата доступа: 11.04.2024

2. Отберём то, что нужно Data mining: как сформировать датасет для машинного обучения [Электронный ресурс]. – Режим доступа: https://bigdataschool.ru/blog/dataset-data-preparation.html - Дата доступа: 11.04.2024

3. Вредные советы по подготовке датасета [Электронный ресурс]. – Режим доступа: https://habr.com/ru/articles/746802/ - Дата доступа: 11.04.2024

4. Реализуем и сравниваем оптимизаторы моделей в глубоком обучении [Электронный ресурс]. – Режим доступа: https://habr.com/ru/companies/skillfactory/articles/525214/ - Дата доступа: 11.04.2024