

Домашнее задание по теме “Распознавание цифр”

Выполнила: Китова Екатерина Денисовна, 324 группа

Постановка задачи:

3* Дополните программу, чтобы можно было рисовать мышкой цифру для распознавания. Проведите эксперименты с рисунками цифр, не входящими в выборку - изменилась ли точность? Проведите эксперименты - существуют ли ситуации, когда человек без труда распознает цифру, а модель стандартно ошибается?

Решение:

Я дополнила исходную программу с сайта(без изменения параметров самой модели) библиотекой `pygame`, чтобы можно было рисовать мышкой. Код находится в файле **digits.py** в этом же архиве.

В изначальном датасете все картинки на белом фоне с черными чернилами, поэтому для облегчения распознавания я сделала аналогичную «доску».

Дополнительные библиотеки: **pygame** (установка через `conda install pygame` или `pip install pygame`)

Для управления окном нужна английская раскладка:

SPACE – распознавание моделью

C – очистка экрана

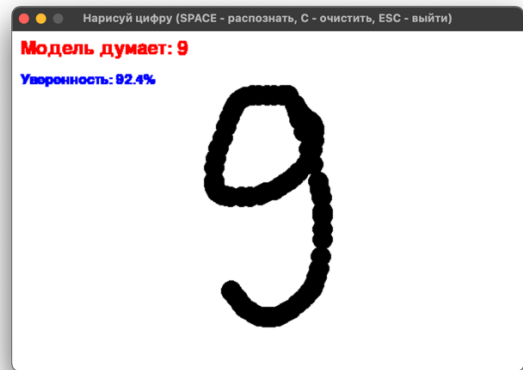
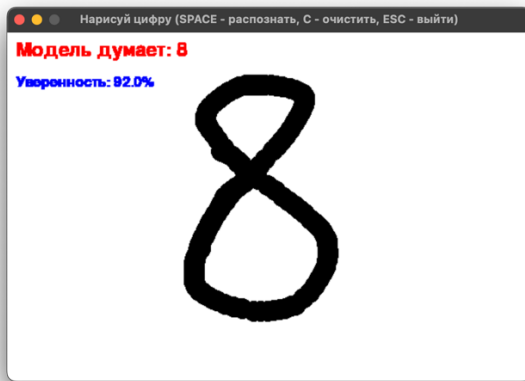
ESC – выход и окончание программы

При нажатие SPACE будет выводиться предсказание модели на картинку и насколько она уверена в предсказании.

Примеры моих рисунков и уверенность модели в результатах(0,1,...,9):





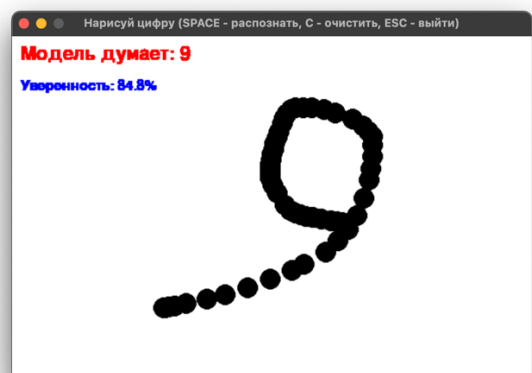


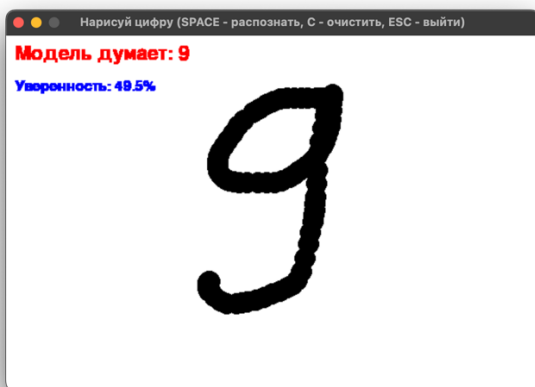
Точность:

Точность будет падать, так как в датасете MNIST картинка уже обрезана по контуру (28x28), отцентрована, нарисована тонкой линией и размер цифры более-менее одинаковый.

У меня же окно 600x400(для удобства рисования), и при ресайзе картинки в формат 28x28(так как модель обучалась на данных формы (28,28,1), поэтому рисунок тоже приводится к такой форме — нормализуется в диапазон [0,1] и дополняется размерностью канала), 4 цифры могут становиться кляксами. Также мы можем ее рисовать НЕ по центру. Из-за особенностей ругате у нас кисть толстая и это тоже может ухудшать точность работы. Если говорить терминами, то распределение обучающих данных не совпадает с распределением данных при инференсе. Была попытка в предобработке централизовать цифру на картинке, но тогда точность ещё больше падала, поэтому оставила лучший вариант - стандартный.

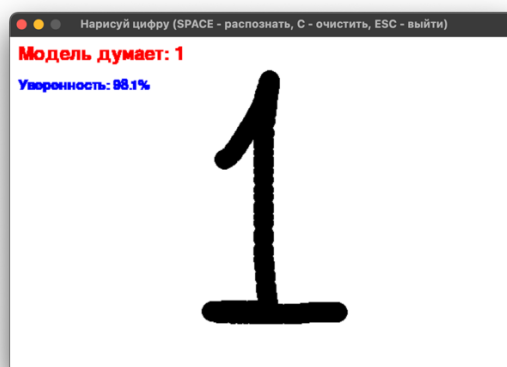
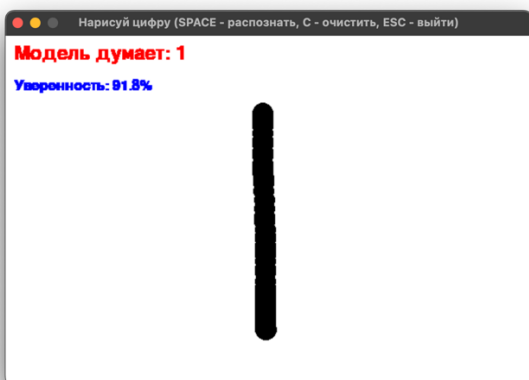
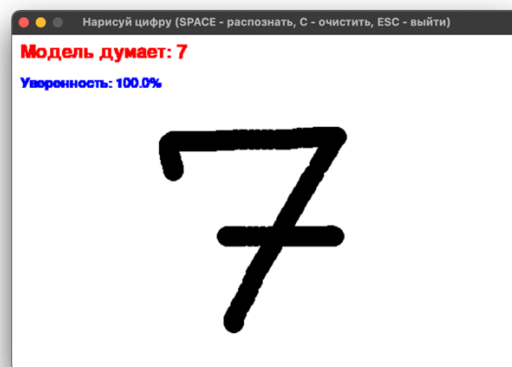
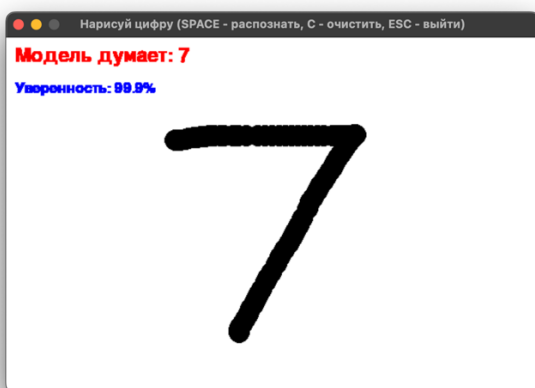
Примеры с 9(модель почему-то очень не хотела ее распознавать, но об этом далее):





Можно заметить, что на некоторых картинках линии недоведены до конца, но при сжатии видимо они убираются. Также что при малейших изменениях точность меняется, то есть с анализом этих картинок модель очень чувствительна к изменениям.

Тесты разного написания одного и того же числа:



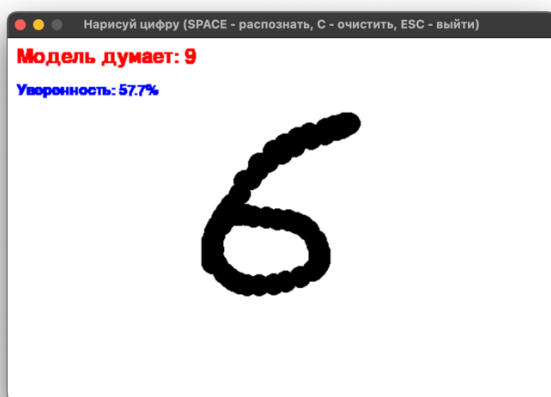


Неудачные распознавания «9»:

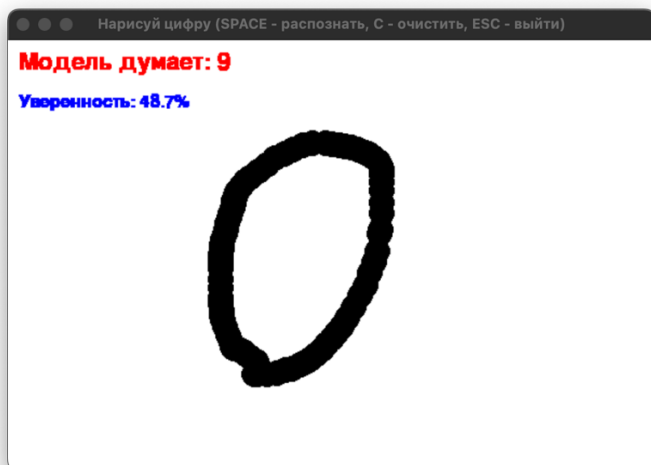


Как мы видим человеку легко распознать, что это девять, но у модели есть проблемы с распознаванием. На примерах выше видно, что модель в двух случаях на 99.4% была уверена, что это 2 или 3. Это происходит из-за выше указанных причин ухудшения точности.

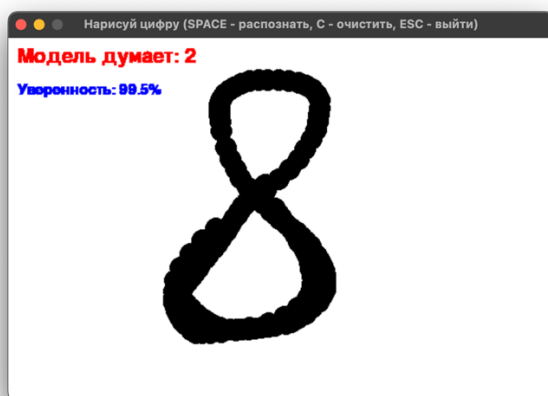
При это, если рисовать четкую «6», то модель будет считать ее за «9», хоть и уверенность примерно 0,57:



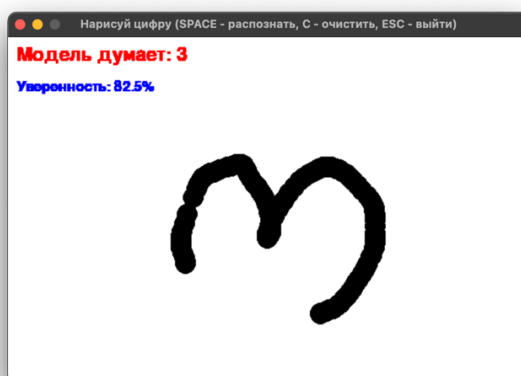
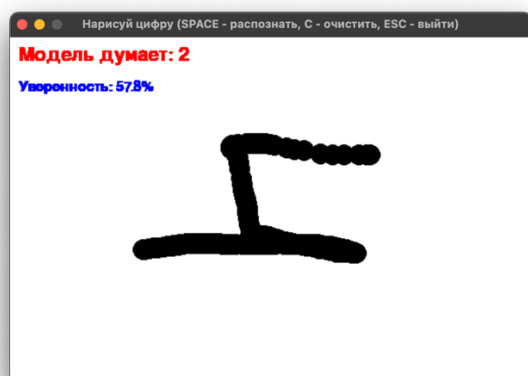
Такая же ситуация иногда происходит с «0», но тут уверенность уже 0,48:

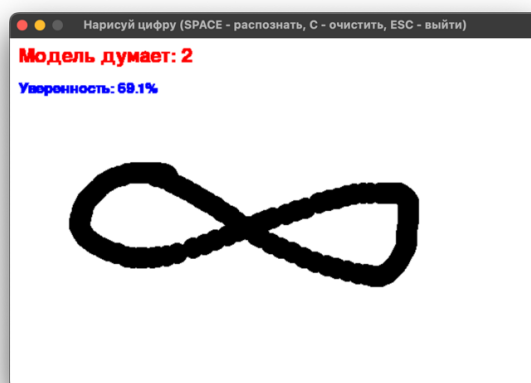


Также иногда возникали путаницы у «8», если рисовать ее чуть шире или толще:



Также я решила попробовать «покрутить» экран и писать перевернутые цифры:





Судя по-всему, когда модель видит штрихи, она в первую очередь подозревает, что это будет четверка, это хорошо заметно на перевернутых 1 и 7, но при этом 4 она посчитала за 2. Хотя верно определила в одном из перевернутых случаев 3 и 0. Отдельно стоит рассмотреть случаи перевернутых 6 и 9, в этом случае даже человеку будет сложно отличить.



Выводы:

Таким образом, простое добавление окна для рисования показало, что модель, обученная на MNIST, чувствительна к предобработке входных изображений, но всё же в большинстве случаев хорошо предсказывает цифру, при аккуратном и точном её написании. Человек легко обобщает разные начертания цифр (перевернутые или разные формы их написания), а модель - нет, потому что она видела только MNIST-подобные изображения. Поэтому при использовании реального ввода нужно либо делать предобработку (обрезка, центрирование, нормализация толщины), либо можно дообучить на небольшом наборе собственных нарисованных цифр, что тоже уменьшит число ошибок.