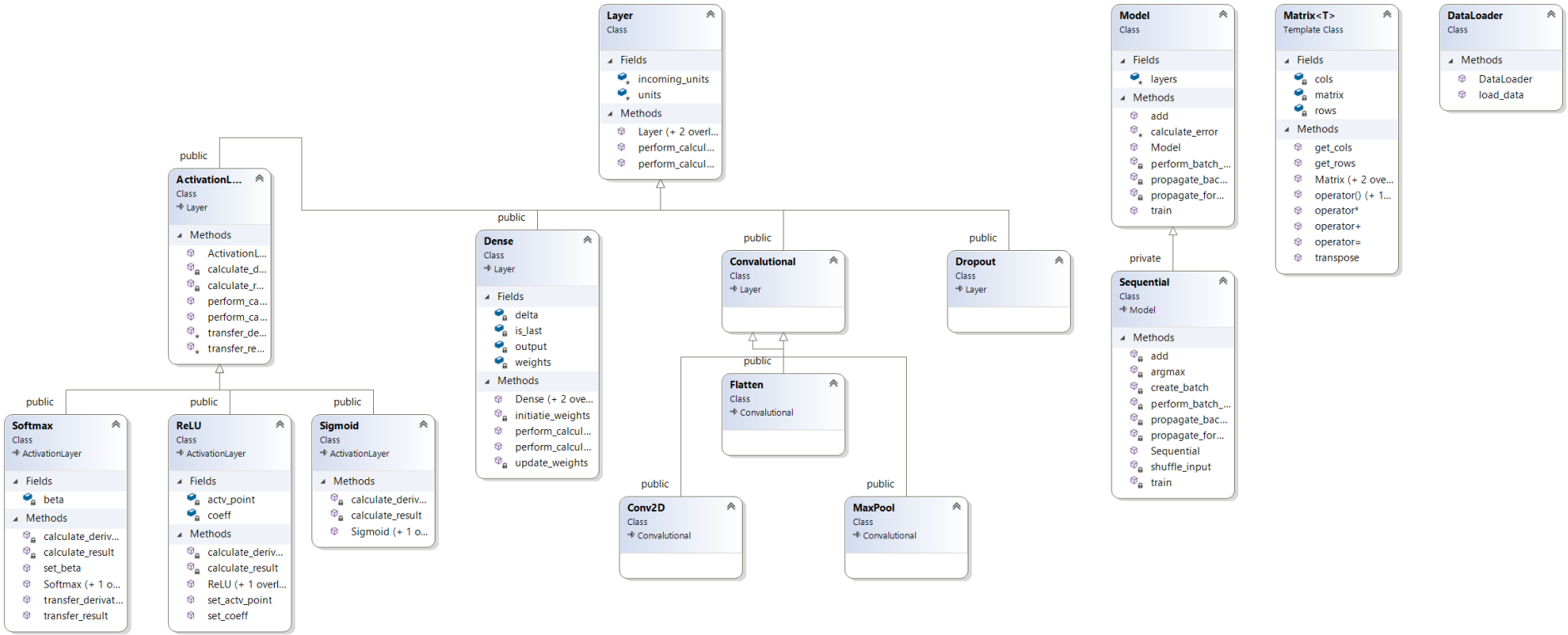
Layer
Class
Fields
incoming_units
units
Methods
Layer (+ 2 overl...
perform_calcul...
perform_calcul...
Model
Class
Fields
layers
Methods
add
calculate_error
Model
perform_batch_...
propagate_bac...
propagate_for...
train
Matrix<T>
Template Class
Fields
cols
matrix
rows
Methods
get_cols
get_rows
Matrix (+ 2 ove...
operator() (+ 1...
operator*
operator+
operator=
transpose
DataLoader
Class
Methods
DataLoader
load_data
public
ActivationL...
Class
Layer
Methods
ActivationL...
calculate_d...
calculate_r...
perform_ca...
perform_ca...
transfer_de...
transfer_re...
public
Dense
Class
Layer
Fields
delta
is_last
output
weights
Methods
Dense (+ 2 ove...
initiatie_weights
perform_calcul...
perform_calcul...
update_weights
public
Convalutional
Class
Layer
public
Dropout
Class
Layer
private
Sequential
Class
Model
Methods
add
argmax
create_batch
perform_batch_...
propagate_bac...
propagate_for...
Sequential
shuffle_input
train
public
Softmax
Class
ActivationLayer
Fields
beta
Methods
calculate_deriv...
calculate_result
set_beta
Softmax (+ 1 o...
transfer_derivat...
transfer_result
public
ReLU
Class
ActivationLayer
Fields
actv_point
coeff
Methods
calculate_deriv...
calculate_result
ReLU (+ 1 overl...
set_actv_point
set_coeff
public
Sigmoid
Class
ActivationLayer
Methods
calculate_deriv...
calculate_result
Sigmoid (+ 1 o...
public
Flatten
Class
Convalutional
public
Conv2D
Class
Convalutional
public
MaxPool
Class
Convalutional

Matrix<T> - klasa generyczna macierzy, w projekcie każdy obiekt przetwarzany przez model sieci jest macierzą (w bibliotece tensorflow jest to tensor). Zawiera metody typowe dla operacji na macierzach, ale głównie używane jest jedynie mnożenie i transpozycja.

DataLoader – klasa zajmująca się wczytywaniem i zapisywaniem danych z/do pliku. Na ten moment gotowe jest wczytywanie danych z pliku csv i zwracane jako std::list<Matrix<float>>. Ewentualnie przewidziany jest zapis i wczytanie modelu.

Model – klasa bazowa modelu, składa się z listy warstw (klasa Layer, dodawanych dzięki metodzie add()), które wykonują odpowiednie obliczenia wewnątrz funkcji modelu train(). Logika tworzenia modelu sieci neuronowej nakazuje zachowanie pewnych założeń co do kolejności i rozmiarów dodawanych warstw. Na ten moment brakuje metody prerdict () służącej do predykcji wyuczonego klasyfikatora.

Sequential – klasa dziedzicąca po klasie Model, to konkretna implementacja założeń modelu.

Layer – klasa bazowa warstwy modelu, zawiera informacje o liczbie sztucznych neuronów wchodzących i wychodzących z warstwy. Zawiera dwie metody wirtualne: perform_calculation_forward i perform_calculation_backwards, które są wykonywane w metodzie train modelu w odpowiedniej kolejności.

Dense – klasa reprezentująca połączenia między sztucznymi neuronami za pomocą macierzy weights. Co istotne, w implementacji metody perform_calculation_backwards dokonuje się również odświeżenie macierzy weigths (w funkcji update_weights).

ActivationLayer – klasa bazowa reprezentująca warstwę aktywacji, wprowadza metodę tranfser_result i transfer_derivative, których konkretna implementacja pozwala na wykonywanie odpowiednich obliczeń podczas propagacji obliczeń w przód i błędu wstecz.

ReLU – klasa wprowadzająca logikę warstwy ReLU w propagacji wyniku w przód i błędu wstecz: https://en.wikipedia.org/wiki/Rectifier_(neural_networks)

Sigmoid - klasa wprowadzająca logikę warstwy Sigmoid w propagacji wyniku w przód i błędu wstecz: https://en.wikipedia.org/wiki/Sigmoid_function

Softmax - klasa wprowadzająca logikę warstwy Softmax w propagacji wyniku w przód i błędu wstecz: https://en.wikipedia.org/wiki/Softmax_function

Convolutional – klasa bazowa warstwy konwolucyjnej. Klas konwolucyjnych używa się na początku listy warstw w celu m.in. redukcji rozmiaru zadania. Te warstwy nie biorą udziału w treningu sieci. Na ten moment brak implementacji, wprowadzę to na sam koniec, ponieważ te warstwy nie są niezbędne do działania klasyfikatora do problemu predykcji liczb.

Conv2D - klasa wprowadzająca logikę warstwy Conv2D w celu zastosowania filtrów: https://www.tensorflow.org/api_docs/python/tf/keras/layers/Conv2D

MaxPool - klasa wprowadzająca logikę warstwy MaxPool – wybór największej wartości dla danego okna: https://www.tensorflow.org/api_docs/python/tf/keras/layers/MaxPool2D

Flatten - klasa wprowadzająca logikę warstwy Flatten w celu spłaszczenia macierzy wejściowej

Dropout - klasa wprowadzająca logikę warstwy Dropout, polegającą na wyborze losowych neuronów które są aktywowane w propagacji obliczeń. Na ten moment brak implementacji.