Optymalną architekturę dla moduły LSTM wybraliśmy porównując wartości MSELoss dla zbioru testowego. Najniższa wartość (uzyskana w dowolnym epochu) wskazywała na najlepszy model. Zmian dokonywaliśmy w parametrach hidden\_size i num\_layers LSTM z biblioteki pytorch. Wszystkie ustawiane przez nas parametry oraz z input\_size mogły być zmieniane w poszukiwaniu najlepszego modelu.

Ostatecznym wynikiem była poniższa klasa:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Model trenowany był wg algorytmu:

Obraz zawierający tekst, zrzut ekranu, Czcionka, oprogramowanie

Opis wygenerowany automatycznieZapobiegliśmy przeuczeniu poprzez zachowanie jedynie modelu o najniższym mean square error (MSE).

Intuicyjnie wiemy, że model jest bardzo słaby. Użycie MSE pozwala przedstawić wysoki wynik, ponieważ nie traktuje wyniku binarnie a pozwala zaliczyć wyniki zbliżone jako sukcesy i pozwala na swobodę w obsłudze wartości ujemnych i niecałkowitych. Najniższy osiągnięty MSE to 7,0323.  
Dla przykładu fragment z „Pana Tadeusza”, gdzie należy dokończyć imię Napoleona we fragmencie „wojna niechybna! kiedy z poselstwem tajemnem tu biegłem, wojsk forpoczty już stały nad niemnem; nap” daje wynik „—c,noezęcpłn—m dswetnłwtreotpcrłddc —r,rtnenrtnóu ciceśuótrzęupśdnziderlplzódatetłó,etwzaoeócuuecau”. Dla wejścia z poza „Pana Tadeusza” „mała małgosia miała mały młyn i małego kotka i małego braciszka i mały samochodzik i małą piłeczkę „ gdzie można spodziewać się kontynuacji małych własności Małgosi uzyskano odpowiedź „óełwoupuęzp wwztwzokwpóoz—enazórcęo rodeadpzotrłpczzzi,wdzepi wrzz,wnnrosrpółt,aęgępgcóódpeacd,zwop”, która nie kończy się na oczekiwane „i”.

Dokładność mierzona jako identyczne wartości wyłącznie części całkowitych dla pełnego wyjścia do pełnego oczekiwanego wyjścia to średnio 0,03667106038161387 a w najlepszym przypadku 0,16161616161616163 dla całego zbioru danych. Dla identycznych warunków, ale liczb zaokrąglonych to maksymalnie 0,16161616161616163 i średnio 0,03582954381086984. Przy porównywaniu tylko ostatnich liter średnia dokładność to 0,03630874893387799 dla ucinania i 0,03688557705256543 dla zaokrąglania liczb. Dekodowanie używa ucinania.

Optymalna architektura TransformerDecoder została wybrana porównując wartości MSELoss dla zbioru testowego. Najniższa wartość (uzyskana w dowolnym epochu) wskazywała na najlepszy model. Zmienialiśmy sugerowany parametr num\_layers i nhead. Wszystkie ustawiane przez nas parametry oraz z input\_size mogły być zmieniane w poszukiwaniu najlepszego modelu.

Ostatecznym wynikiem była poniższa klasa: Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Model trenowany był zgodnie z algorytmem:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Zapobiegliśmy przeuczeniu poprzez zachowanie jedynie modelu o najniższym mean square error (MSE).

Mimo lepszych wyników, model ten również pozostawia wiele do życzenia. Dla tych samych wejść co LSTM odpowiada „ię gepwszewiz arae óiwrn oętwdeewtrwo—aopniuwaenriptziaoip ęę wswrwgzerpepz caenłazwóraezuowdnc, ż” na fragment „Pana Tadeusza” i „sięa geezgw e e zępwóewioztreezwwnnwwat nwrecrgpwdaaoawpęięa ęeezgwrpwepoazęióuwówuręppcwgzo,łd,e” na listę rzeczy Małgosi.

Żaden z utworzonych przez nas modeli nie jest użyteczny. Jednak dzięki swojej bazie polskiej literatury mogą być wykorzystane do generowania unikalnych i bezpiecznych haseł.  
Niska wartość MSE dla modelu TransformerDecoder może nie być w pełni miarodajna ponieważ ilość danych na jakich był trenowany i testowany jest mniejsza niż dla modelu LSTM (dla pełnego zestawu danych 1 epoch zajmował ponad 2 godziny).