Rozpoznawanie człowieka metodami biometrii

Sprawozdanie 2.

**Identyfikacja użytkownika na podstawie dynamiki pisania na klawiaturze**

Autor:

Kacper Trojanowski

Warszawa, Maj 2017

# Spis treści



# 1 Wstęp

Dynamika pisania na klawiaturze jest cechą człowieka, którą można wykorzystać do jego identyfikacji lub też weryfikacji jego tożsamości. Każdy użytkownik komputera posiada umiejętność pisania na klawiaturze, lecz sposób pisania jest charakterystyczny dla pojedynczego osobnika - sposób trzymania dłoni na klawiaturze, sposób wprowadzania wielkich liter, znaków specjalnych i przede wszystkim rytm i szybkość pisania dają nie mniej informacji o użytkowniku niż tradycyjny podpis dzięki czemu, używając metod uczenia maszynowego na podstawie wcześniej zebranych próbek, możemy stwierdzić, kto używa klawiatury. Zdecydowanym atutem biometrii pisania na klawiaturze jest prostota i wygoda użytkownika - nie jest on zmuszany do skanowania linii papilarnych, tęczówki czy innych cech fizjologicznych. Zbieranie informacji odbywa się w trakcie zwyczajnego korzystania z komputera. Z drugiej strony istnieje wiele czynników, które tę identyfikację mogą zaburzyć takie jak kontuzje dłoni, pośpiech, czy zmiana samej klawiatury. Jednak mimo tych niedoskonałości systemy biometryczne oparte o dynamikę pisania na klawia-

turze mają swoje zastosowanie w praktyce, jako sposób autoryzacji razem z innymi metodami (2 factor authentication), autoryzacja ciągła pozwalająca na sprawdzenie, czy intruz nie przejął np. otwartej sesji z systemem, lub też do profilowania użytkowników strony [1].
Niniejsze sprawozdanie opisuje eksperyment sprawdzający wyniki identyfikacji użytkowników na podstawie zebranych wcześniej próbek pisma. W celu realizacji zadania otrzymane dane należało w odpowiedni sposób przetworzyć, przeprowadzić ekstrakcję cech, a następnie zastosować wybrany model uczenia maszynowego.

# 2 Opis techniczny

W celu realizacji zadania korzystałem ze środowiska *Anaconda* - opartego na języku *Python 2.6*. W celu preprocessingu danych i przechowywania wektorów cech zastosowałem silnik bazy danych *SQLite 3*. Do implementacji sieci neuronowej wykorzystałem bibliotekę *Keras* opartej o *Theano*. Do przygotowania danych w celu nauki sieci wykorzystałem bibliotekę *scikit-learn*.

# 3 Zebrane dane

Dane wykorzystane do przeprowadzenia eksperymentów pochodzą od 153 użytkowników, którzy poprzez przeglądarkę internetową wielokrotnie wpisywali zadane słowo - **.tie5Roanl**. Aplikacja webowa zbierająca dane zapisywała zdarzenia klawiszy klawiatury (key up, key down), kod przycisku, czas zdarzenia, oraz pozycję kursora. Poprawnie wpisana próbka powinna zawierać 24 zdarzenia klawiatury - przyciśnięcie i podniesienie 10 widocznych znaków, klawisza SHIFT w celu wpisania wielkiej litery R, klawisza ENTER w celu akceptacji próbki. Początkowo baza zawierała 29255 takich próbek. Zapisane były one jako łańcuchy znaków, każda próbka w jednym wierszu tabeli SQL.

# 4 Preprocessing

Pierwszym krokiem w realizacji zadania było przeprowadzenie preprocessingu danych, czyli filtracja błędnych próbek, niezgodnych z zadanym wzorcem, oraz odrzucenie niedoskonałości zapisu. W pierwszym kroku podzieliłem dostępne próbki na zdarzenia co zapisałem w tabeli SQL - czyli każda próbka miała ok. 24 zdarzenia zapisane w tabeli, które wczytane razem dawały wszystkie zdarzenia key up&down danej próbki. Taki sposób zapisu próbek ułatwił ich późniejsze przetwarzanie.

## 4.1 Usunięcie powtarzających się zdarzeń

Kolejnym krokiem, dla każdej próbki było usunięcie nadmiarowych zdarzeń, które nie powodowały wprowadzenia błędnego wzorca, lecz były wynikiem charakterystycznego wprowadzania tekstu przez danego użytkownika. Gdy użytkownik dłużej przytrzymał jeden klawisz, przeglądarka zapisywała zdarzenie key down wielokrotnie, co nie było istotne z punktu widzenia działania naszego systemu. Stąd też nadmiarowe zdarzenia zostały usuwane, a czas nacisku na klawisz, był liczony od pierwszego wciśnięcia do podniesienia klawisza.

## 4.2 Filtrowanie nieprawidłowego wzorca

Po wstępnym przekształceniu próbki odfiltrowane zostały te, które nie odpowiadały wzorcowi, czyli nie spełniały początkowo trzech warunków:

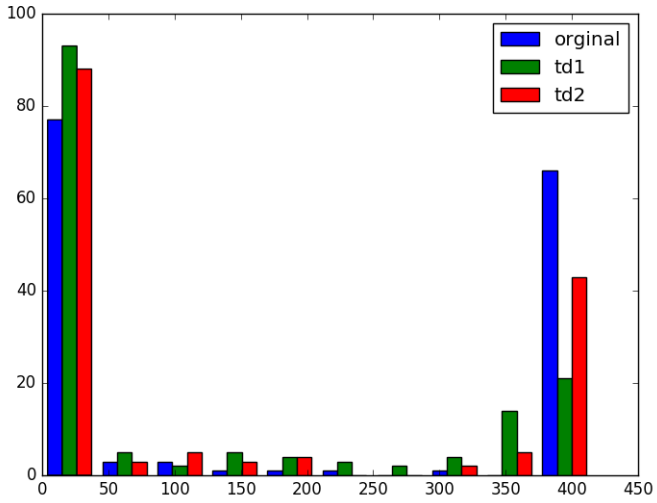
- Ilość zdarzeń nadal była różna od 24 - usunęło to próbki błędne - niepełne, lub z dodatkowymi znakami. Zostały tutaj usunięte również próbki, które mogły poprawnie przedstawiać wzorzec, lecz zostały wpisane w sposób znacząco odmienny od większości próbek, czyli na przykład z użyciem przycisku CAPS LOCK, innych klawiszy specjalnych, korekcji błędów klawiszem BACKSPACE. Mimo, że próbki te mogły przedstawiać poprawne wzorce, mogłyby zaburzać wyniki modelu, gdyż były wynikiem błędu, lub świadomej manipulacji użytkownika.

- Kolejność uderzanych klawiszy była odpowiednia - sprawdzone zostały przeze mnie

kody wciskanych przycisków, oraz czy ich kolejność była poprawna. Sprawdzenie to opierało się jedynie o zdarzenia key down.

- Kolejność zdarzeń up & down - sprawdzona została kolejność zdarzeń key up & key down przy założeniu, że poza klawiszem SHIFT, bezpośrednio po naciśnięciu na klawisz następowało jego zwolnienie.

Po obserwacji wyników otrzymanych w ten sposób, zaobserwowałem, że dla wielu użytkowników, którzy wprowadzili niemałą liczbę próbek wiele z nich zostało odfiltrowanych. Po analizie takich przypadków zauważyłem, że cechą charakterystyczną pisania niektórych użytkowników jest, nie jakby się wydawało naturalne zwalnianie klawisza po jego naciśnięciu, ale zwolnienie go, po naciśnięciu kolejnego klawisza. Zachowanie takie, można zaobserwować zwłaszcza w późniejszych próbach danego użytkownika, gdy wpisywał już wzorzec wielokrotnie.




Rysunek 1: Histogram liczby próbek przypadających na danego użytkownika dla zestawu oryginalnego, td1 - po zastosowaniu wszystkich trzech warunków, td2 - po rezygnacji z warunku kolejności zdarzeń up & down. Można zauważyć, że większość użytkowników, albo zrezygnowało szybko po wpisaniu małej ilości próbek, lub też wpisywała dyżą ilość próbek (ok. 400)

Ostatecznie, do nauki modelu uczenia maszynowego wykorzystano użytkowników, którzy wpisali niemniej niż 5 próbek spełniających pierwsze dwa warunki, co dało ogółem 122 użytkowników.

# 5 Ekstrakcja cech

Po przetworzeniu danych na podstawie próbek wyznaczone zostały wektory cech dla każdej próbki. Pojedynczy wektor cech składał się z 24 wielkości, które były *flight time* i *dwell time*. Pierwszy z nich to czas pomiędzy podniesieniem poprzedniego przycisku, a wciśnięciem kolejnego, natomiast *dwell time* to czas w którym dany klawisz był wciśnięty. *Dwell time* przyjmuje wartości całkowite dodatnie, natomiast *flight time* po odrzuceniu warunku o kolejności zdarzeń up & down może przyjmować wartości ujemne (z uwzględnieniem tego warunku wartości ujemne wystąpiły by tylko dla klawisza SHIFT). W celu przyspieszenia eksperymentów, wektory cech zostały przeze mnie zapisane w kolejnej tabeli SQL, dzięki czemu testując modele nie było konieczności ponownego przeprowadzania preprocessingu i ekstrakcji cech.

# 6 Nauka i testowanie modelu

Zastosowanym przeze mnie modelem została sieć neuronowa MLP (Multi-Layer Perceptron). Plan przeprowadzanych eksperymentów zakładał przetestowanie wartości parametrów Learning Rate oraz Momentum dla zastosowanego algorytmu uczenia sieci - *Stochastic Gradient Descent*. Do porównywania wyników używałem metody walidacji krzyżowej - k-krotnej walidacji, o wartości k=5. Podział na zbiory uczące i testowe został dokonany przez bibliotekę *scikit-learn*. Uwzględniona została przeze mnie opcja zachowania proporcji użytkowników we wszystkich podzbiorach (algorytm StratifiedKFold), przez co minimalną ilością próbek dla danego użytkownika było 5. Użytkownicy o mniejszej ilości próbek zostali odrzuceni. Sieć neuronowa posiadała wejście równe oczywiście wielkości wektora cech - 24, w kolejnych dwóch warstwach znajdowały się 64 neurony, natomiast warstwa wyjściowa posiada liczbę neuronów równą ilości użytkowników - w przypadku

przeprowadzonych przeze mnie eksperymentów było to 122. Warstwy ukryte posiadały funkcję aktywacji ReLU, natomiast warstwa wyjściowa funkcję SoftMax. W celu nauki sieci neuronowej identyfikatory użytkowników zostały zakodowane jako liczby z przedziału $[0, 122)$, natomiast każda z cech przeskalowana jest do wartości $[-1, 1]$. Miarą jakości modelu była celność identyfikacji użytkowników czyli stosunek poprawnie zidentyfikowanych użytkowników do wszystkich użytkowników w zbiorze testowym, określone jako $ACC = \frac{successcount}{testdatasize}$.

| Learning Rate | Momentum | Śr. ACC |
|---|---|---|
| 0.05 | 0.9 | 0.75 |
| 0.05 | 0.45 | 0.74 |
| 0.05 | 0.2 | 0.72 |
| 0.01 | 0.9 | 0.76 |
| 0.01 | 0.45 | 0.63 |
| 0.01 | 0.2 | 0.57 |
| 0.005 | 0.9 | 0.73 |
| 0.005 | 0.45 | 0.52 |
| 0.005 | 0.2 | 0.44 |

Jak widać w powyższej najwyższy wynik osiągnięty został dla $LearningRate = 0.01$ oraz $Momentum = 0.9$. Uzyskana celność wyniosła $0.76$. Dla tych parametrów zwiększyłem liczbę epok do 200 i przeprowadziłem eksperyment 10-krotnie, aby poznać jak stabilny jest ten model i uzyskać uśrednione wyniki. Wyniki przedstawiam poniżej:

| Id. | Śr. ACC |
|---|---|
| 1 | 0.77616714035 |
| 2 | 0.776082429887 |
| 3 | 0.771736187829 |
| 4 | 0.772650744363 |
| 5 | 0.775102647526 |
| 6 | 0.77039611433 |
| 7 | 0.777368662132 |
| 8 | 0.781217931659 |
| 9 | 0.779519500063 |
| 10 | 0.777875771332 |

Ostatecznie średnia skuteczność tej metody wyniosła $ACC = 77.58\%$. Odchylenie standardowe dla 10 prób wyniosło $\sigma = 0.0034$.

# 7 Wnioski

Biometryczna identyfikacja człowieka na podstawie dynamiki pisania na klawiaturze w sposób przedstawiony w tym sprawozdaniu nie daje wyników na tyle doskonałych aby być używaną jako samodzielna metoda autoryzacji użytkownika do systemu. Dodatkowo jest podatna na wiele czynników, mogących zaburzyć wyniki identyfikacji użytkownika. Jednak uzyskany wynik $77.58\%$ poprawności w identyfikacji 122 użytkowników pokazuje, że metoda ta może być z powodzeniem stosowana jako dodatkowa forma sprawdzenia tożsamości użytkownika obok tradycyjnych metod, takich jak np. hasło, lub też w systemach o mniejszych wymogach bezpieczeństwa, lecz wymagających braku ingerencji w działania użytkownika - jak na przykład zbieranie danych o użytkownikach aplikacji internetowej. Dodatkowo, jest to cecha behawioralna, której podrobienie nie jest trywialnym zadaniem [2]. Potencjalnym polem do poprawy osiągniętych wyników byłoby rozszerzenie wektora cech o dodatkowe cechy pozyskane z danych - jak na przykład użycie przycisku CAPSLOCK zamiast SHIFT oraz uwzględnienie wprowadzania poprawek do próbki przyciskiem BACKSPACE lub DELETE. Dodatkowo w ten sposób uniknęlibyśmy

potrzeby odrzucenia sporej części danych.

## Literatura

[1] *typingDNA*, https://typingdna.com/

[2] Fred Erlend N. Rundhaug , *Keystroke dynamics Can attackers learn someone's typing characteristics*

[3] *Scikit-learn: Machine Learning in Python*, Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P. and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and Cournapeau, D. and Brucher, M. and Perrot, M. and Duchesnay, E.

[4] *Keras*, Chollet, François and others