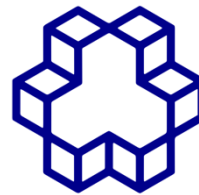


به نام خدا



گروه پژوهشی ایک

دانشگاه صنعتی خواجه نصیرالدین طوسی  
دانشکده برق



دانشگاه صنعتی خواجه نصیرالدین طوسی

مبانی سیستم های هوشمند

گزارشکار پروژه پایانی

پارسا رشتیان

۴۰۰۱۰۸۲۳

استاد : آقای دکتر مهدی علیاری

دی ۱۴۰۳

فهرست مطالب

مقدمه

روش ها و مدل ها

چالش ها و مشکلات موجود

متدلوژی

نتایج و تحلیل آن ها

نتیجه گیری

## مقدمه

\*هدف پروژه : هدف اصلی از این پروژه، شناسایی و تشخیص اثر انگشت افراد با استفاده از یک مدل CNN و در نهایت مقایسه‌ی آن با یک شبکه‌ی عصبی مبتنی بر الگوریتم همینگ (Hamming NN) است. این کار با هدف مقایسه عملکرد هر کدام از این مدل‌ها بر روی داده‌های واقعی انجام شده است. در دنیای امروز سیستم‌های تشخیص اثر انگشت یکی از مهم‌ترین فناوری‌های بیومتریک هستند که امروزه در بسیاری از صنایع و کاربردهای امنیتی استفاده می‌شوند. این سیستم به دلیل دقت بالا، هزینه‌ی نسبتاً کم و سهولت استفاده، جایگزین روش‌های سنتی احراز هویت مانند رمزهای عبور و کارت‌های شناسایی شده‌اند. در ادامه برخی از مهم‌ترین کاربردهای سیستم‌های اثر انگشت در دنیای واقعی را بررسی می‌کنیم:

۱. امنیت و کنترل دسترسی
۲. بانکداری و پرداخت‌های دیجیتال
۳. شناسایی مجرمان
۴. دستگاه‌های هوشمند و اینترنت اشیا

## روش‌ها و مدل‌ها

در این پروژه از دو روش یادگیری برای شناسایی اثر انگشت استفاده شده است:

۱. شبکه‌ی عصبی کانولوشنی (Convolutional Neural Network) :

CNN یک مدل یادگیری عمیق است که در شناسایی تصاویر بسیار خوب عمل می‌کند. این مدل با استفاده از چندین لایه، ویژگی‌های تصاویر را یاد می‌گیرد و در نهایت آن را طبقه‌بندی می‌کند.

مراحل کار CNN :

۱. لایه‌ی کانولوشن: یک فیلتر کوچک روی تصویر حرکت می‌کند و الگوهای مهم (لبه‌ها، بافت‌ها و اشکال) را تشخیص می‌دهد. این فیلترها نقش استخراج ویژگی‌های تصویر را دارند.

فرمول محاسبه‌ی کانولوشن:

$$Z_{i,j}^l = \sum_m \sum_n X_{i-m,j-n}^{(l-1)} \cdot W_{m,n}^{(l)} + b^{(l)}$$

X تصویر ورودی / W فیلتر / b بایاس

فیلترهای معروفی که در CNN استفاده می‌شوند:

۲. تابع فعال‌سازی (ReLU – Rectified Linear Unit)

$$ReLU(x) = \max(0, x)$$

تمام مقادیر منفی را صفر و فقط مقادیر مثبت را نگه می‌دارد.

۳. لایه‌ی تجمعی (Pooling Layer) برای کاهش ابعاد

این لایه باعث کاهش اندازه‌ی ویژگی‌ها و افزایش سرعت پردازش می‌شود. Max Pooling متداول‌ترین روش است که مقدار بیشترین پیکسل را نگه می‌دارد.

$$Z_{i,j} = \max(X_{i+m,j+n}), \forall m, n \in \text{window}$$

۴. لایه‌ی کاملاً متصل (FC Layer – Fully Connected Layer)

ویژگی‌های استخراج‌شده به یک خروجی نهایی تبدیل می‌شوند.

تابع Softmax برای پیش‌بینی کلاس‌ها:

$$P(y_i) = \frac{e^{z_i}}{\sum_{j=1}^N e^{z_j}}$$

این تابع احتمال هر کلاس را محاسبه می‌کند.

- مزایا: دقت بالا / توانایی شناسایی الگوهای پیچیده
- معایب: نیاز به داده‌های زیاد / زمان بسیار زیاد برای آموزش مدل

## ۲. شبکه‌ی عصبی همینگ (Hamming Neural Network)

شبکه‌ی همینگ یک روش تطبیق الگو است که بر خلاف CNN با یادگیری ویژگی‌ها جلو نمی‌رود بلکه شباهت آن را با تصاویر ذخیره‌شده مقایسه می‌کند.

مراحل کار Hamming NN :

۱. تبدیل تصویر به ماتریسی از مقادیر باینری - هر پیکسل بر اساس شدت رنگ سفید (1) یا سیاه (0) در نظر گرفته می‌شود.

$$B_{i,j} = \begin{cases} 1 \\ -1 \end{cases}$$

۲. محاسبه‌ی فاصله‌ی همینگ

فاصله‌ی بین تصویر ورودی و تصاویر ذخیره‌شده محاسبه می‌شود.

کمترین مقدار فاصله نشان‌دهنده بیشترین شباهت است.

فرمول فاصله همینگ:

$$d_h(A, B) = \frac{\sum_{j=1}^N |A_i - B_i|}{N}$$

۳. انتخاب نزدیک‌ترین تطابق:

تصویری که کمترین فاصله‌ی همینگ را دارد به عنوان خروجی انتخاب می‌شود.

- مزایا : بسیار سریع / بدون نیاز به آموزش
- معایب : دقت کمتر / حساسیت بالا به نویز

ویژگی	CNN	Hamming NN
نوع یادگیری	یادگیری عمیق	تطبیق الگو
نیاز به آموزش	بله	خیر
دقت	بالا	متوسط
سرعت پردازش	کند	بسیار سریع

## چالش‌ها و مشکلات موجود

۱ - مشکلات مربوط به داده: یکی از اولین چالش‌هایی که با آن مواجه شدم، کیفیت و حجم داده‌های موجود بود. در ابتدا، دیتاستی که انتخاب کرده بودم شامل ۸۰ تصویر بود که فقط ۸ انگشت از ۱۰ نفر را در بر می‌گرفت. این حجم پایین داده باعث شد که مدل CNN نتواند الگوهای کافی را استخراج کند و در نتیجه، مدل عملکرد ضعیفی داشت.

برای رفع این مشکل، از دیتاستی بزرگ‌تر و غنی‌تر استفاده کردم که شامل ۵۲۰ تصویر از ۷ فرد مختلف بود. تفاوت این دیتاست با نمونه‌ی قبلی در این بود که برای هر انگشت، ۸ نمونه مختلف وجود داشت. این ویژگی به شبکه‌ی عصبی کمک می‌کرد که بهتر بتواند ویژگی‌های پایدار اثر انگشت را یاد بگیرد.

با این حال، همچنان حجم داده برای مدلی مثل CNN نسبتاً کوچک بود و نمی‌توانست به اندازه‌ی کافی روی الگوهای متنوع یادگیری کند. برای حل این مشکل، از تکنیک‌های داده‌افزایی (Augmentation) استفاده کردم. با اعمال تغییرات کوچک روی تصاویر اصلی (مانند چرخش، نویز، تغییر روشنایی و کنتراست)، توانستم تعداد داده‌های آموزشی را افزایش دهم و تنوع بیشتری به مدل بدهم. این روش کمک کرد تا شبکه عصبی بتواند پایداری بیشتری در برابر تغییرات تصویر داشته باشد و دقت تشخیص افزایش یابد.

## ۲ - مشکلات مربوط به مدل‌های یادگیری

مدل CNN با داده‌های محدودی که در اختیار داشتیم نتوانست ویژگی‌ها را به خوبی استخراج کند. به همین دلیل، تصمیم گرفتیم که طبقه‌بندی را برای این مدل به تعداد افراد موجود در دیتاست محدود کنیم. به عبارت دیگر، هر انگشت از فرد اول به `Person_0`، هر انگشت از فرد دوم به `Person_1` و به همین ترتیب نسبت داده شد. این رویکرد باعث شد که مدل داده‌های بیشتری برای هر کلاس داشته باشد و بتواند ویژگی‌های مربوط به هر فرد را بهتر یاد بگیرد.

شبکه Hamming NN به دلیل ویژگی‌های تطبیقی‌اش که از مقایسه‌ی تک‌تک پیکسل‌ها برای شناسایی استفاده می‌کند، به نویز و تغییرات کوچک در تصاویر حساس است. این ویژگی

باعث می‌شود که در تصاویر نویزی خطاهایی در شناسایی ایجاد شود. راهکار پیشنهادی برای بهبود این مشکل، استفاده از داده‌های نویزی برای آموزش مدل بوده که متأسفانه به دلیل محدودیت‌های زمانی و منابع نتواستم آن را پیاده‌سازی کنم.

### ۳ - مشکلات مربوط به سرعت پردازش:

زمان آموزش مدل CNN بسیار طولانی بود، و به دلیل نیاز به آزمایش حالت‌های مختلف برای رسیدن به نتیجه‌ی بهینه، هر بار که شبکه CNN آموزش می‌دید، ساعت‌ها زمان از من می‌گرفت. در چنین شرایطی، استفاده از معماری‌های آماده مانند ResNet یا MobileNet که سبک‌تر و سریع‌تر هستند، می‌توانست به صرفه‌جویی در زمان کمک کند. با این حال، من تصمیم گرفتم که همچنان از یک مدل CNN ساده استفاده کنم.

درمورد همین‌گ هم می‌توان گفت پردازش کندی در هنگام تست تصاویر جدید داشتیم. مقایسه‌ی هر تصویر جدید با کل دیتابیس در همین‌گ زمان‌بر است.

### ۴ - چالش کلاس unknown

در مدل CNN، یکی از چالش‌های مهم این بود که مدل بتواند اثر انگشت‌هایی که قبلاً ندیده است را نیز شناسایی کند و از اختصاص دادن آن‌ها به افراد اشتباه جلوگیری کند. برای این منظور، یک کلاس جدید به نام "Unknown" به مدل اضافه کردیم که بتواند اثر انگشت‌های ناشناس را در دسته‌ای جداگانه قرار دهد.

#### چگونه کلاس "Unknown" ساخته شد؟

جمع‌آوری تصاویر ناشناس:

در مجموعه داده‌ها، تصاویری را که به هیچ یک از افراد حاضر در دیتاست تعلق نداشتند، به عنوان نمونه‌های کلاس "Unknown" انتخاب کردیم. این تصاویر شامل موارد زیر بودند:

- اثر انگشت‌هایی که از دیتاست‌های دیگر انتخاب شدند و مدل قبلاً آن‌ها را ندیده بود.
- تصاویر غیر مرتبط مانند نویز، تصاویر دست، و حتی تصاویر حیوانات که مدل نباید آن‌ها را به عنوان اثر انگشت شناسایی کند.

اضافه کردن کلاس هشتم به مدل:

برای اینکه CNN بتواند این کلاس را تشخیص دهد، تعداد کلاس‌های مدل را از ۷ به ۸ افزایش دادیم. این کلاس شامل تمامی تصاویر ناشناخته‌ای بود که مدل باید یاد می‌گرفت که آن‌ها را در هیچ‌یک از ۷ فرد موجود دسته‌بندی نکند. به این ترتیب، مدل یاد گرفت که اگر به اندازه‌ی کافی مطمئن نبود، یک تصویر را به عنوان "Unknown" تشخیص دهد

استفاده از Confidence Threshold:

برای افزایش دقت شناسایی اثر انگشت‌های ناشناس، آستانه‌ی اطمینان (Confidence Threshold) تنظیم شد.

اگر مدل CNN با اطمینان بالایی (مثلاً ۸۰٪ به بالا) یک تصویر را به یک کلاس مشخص نسبت می‌داد، آن را همان کلاس می‌پذیرفت.

اما اگر میزان اطمینان کمتر از حد تعیین‌شده بود، مدل تصویر را در کلاس "Unknown" قرار می‌داد.

چرا این کار مهم بود؟

بدون اضافه کردن کلاس "Unknown"، مدل مجبور بود هر اثر انگشت ورودی را به یکی از افراد شناخته‌شده در دیتاست نسبت دهد. این موضوع باعث خطاهای فاحش در شناسایی می‌شد. اما با افزودن این کلاس، مدل یاد گرفت که تصاویر جدید و اثر انگشت‌های ناشناس را در دسته‌ی جداگانه‌ای قرار دهد و از تخصیص نادرست جلوگیری کند.

۵ - مقایسه دو مدل با هم :

مدل Hamming NN در تشخیص اثر انگشت‌ها هیچ مشکلی نداشت و توانست نام فایل‌های مورد نظر را به درستی پیش‌بینی کند. با این حال، تفاوت اصلی این مدل با CNN در نحوه‌ی دسته‌بندی کلاس‌ها بود.

در Hamming NN، هر تصویر ورودی یک کلاس منحصر به فرد دریافت می‌کرد، به این معنا که هر اثر انگشت به‌طور مستقل شناسایی می‌شد و مدل به جای دسته‌بندی کلی افراد، هر فایل تصویری را به عنوان یک هویت مجزا در نظر می‌گرفت. در مقابل، مدل CNN دارای ۸ کلاس کلی بود که شامل ۷ فرد مختلف و یک کلاس "Unknown" برای تصاویر ناشناس می‌شد.

برای اینکه بتوان مقایسه‌ی عادلانه‌ای بین این دو مدل انجام داد، تصمیم گرفتیم که خروجی‌های Hamming NN را به ۸ کلاس محدود کنیم. به این ترتیب، اگر مدل Hamming NN یک اثر انگشت را متعلق به فردی با شناسه‌ی "۰۱۲" شناسایی می‌کرد، آن را معادل "Person\_0" در CNN در نظر گرفتیم. این کار به ما امکان داد تا دقت مدل‌ها را با یک معیار مشترک ارزیابی کنیم.

## متدلوژی

در این پروژه، ما از دو روش مختلف برای شناسایی اثر انگشت استفاده کرده‌ایم:

۱. شبکه‌ی عصبی پیچشی (CNN) برای یادگیری ویژگی‌های اثر انگشت و پیش‌بینی افراد
۲. شبکه‌ی عصبی همینگ (Hamming NN) برای شناسایی اثر انگشت بر اساس مقایسه‌ی الگوهای دودویی (Binary Patterns)

### ۱ - آماده‌سازی داده‌ها (Preprocessing Data)

آماده‌سازی داده‌ها شامل مراحل زیر است:

خواندن تصاویر از دیتاست موجود

تبدیل تصاویر به فرمت PNG برای اطمینان از سازگاری پردازش‌ها

ساخت دایرکتوری‌های آموزش و قرار دادن تصاویر مربوط به هر شخص در فولدر مشخص

افزودن کلاس "Unknown" که شامل تصاویر غیر مرتبط یا اثر انگشت‌های جدید برای بهبود شناسایی ناشناخته‌ها

اعمال Data Augmentation برای افزایش تنوع داده‌ها و بهبود عملکرد مدل

### ۲ - معماری مدل CNN

مدل شامل ۴ لایه کانولوشنی (Convolutional Layers) بود که ویژگی‌های مختلف اثر انگشت را استخراج می‌کرد.

پس از هر لایه کانولوشنی، از Batch Normalization و MaxPooling برای کاهش ابعاد استفاده شد.

در انتهای شبکه، لایه‌های متصل کامل (Fully Connected) برای دسته‌بندی نهایی قرار گرفتند.



- 1 Conv2D (64 فیلتر) + Batch Normalization + MaxPooling
- 2 Conv2D (128 فیلتر) + Batch Normalization + MaxPooling
- 3 Conv2D (256 فیلتر) + Batch Normalization + MaxPooling
- 4 Conv2D (512 فیلتر) + Batch Normalization + MaxPooling
- 5 Flatten + Dense (1024 نورون)
- 6 Dense (512 نورون)
- 7 Dense ۸ کلاس خروجی

تابع هزینه: Sparse Categorical Crossentropy

بهینه‌ساز: Adam Optimizer با یادگیری ۰.۰۰۱

معیار ارزیابی: Accuracy

### ۳- اورفیت کردن مدل روی دیتای موجود:

چرا مدل را روی داده‌های موجود اورفیت کردیم؟

در این پروژه، هدف ما پیش‌بینی اثر انگشت‌های جدید نبود، بلکه صرفاً تشخیص اثر انگشت‌های شناخته‌شده از دیتاست بود.

برای همین، تمرکز روی حفظ دقت بالا برای نمونه‌های دیتاست موجود بود و مدل نیازی به تعمیم روی اثر انگشت‌های جدید نداشت.

این باعث شد که مدل برای دیتای موجود، به صورت کاملاً بهینه یادگیری انجام دهد.

چگونه اورفیت را کنترل کردیم؟

از Early Stopping برای متوقف کردن یادگیری در نقطه بهینه استفاده شد. (البته در مدل نهایی که ترین شد با استفاده از نتایجی که قبلاً با استفاده از early stopping بدست آمده بود عدد ۵۰ را برای epoch ها قرار دادم تا نتیجه‌ی مطلوب را بدست آورم.)

از Dropout (برای کاهش بیش‌برازش) استفاده نشد، زیرا هدف اصلی حفظ دقت بالا برای داده‌های موجود بود.

کلاس Unknown اضافه شد تا مدل در مواجهه با داده‌های جدید، بتواند آن‌ها را از اثر انگشت‌های شناخته‌شده تفکیک کند.

## ۴ - مدل Hamming NN و نحوه تنظیم آن

چرا مدل Hamming NN را انتخاب کردیم؟

این مدل بر اساس مقایسه‌ی مستقیم اثر انگشت‌های ذخیره‌شده و ورودی کار می‌کند.

بر خلاف CNN، نیازی به یادگیری ندارد و مستقیماً بر اساس محاسبه‌ی فاصله همینگ بین پیکسل‌ها، نزدیک‌ترین اثر انگشت را پیدا می‌کند.

سرعت پردازش بسیار بالایی دارد و برای دیتاست‌های کوچک عملکرد بسیار خوبی نشان می‌دهد.

نحوه‌ی کار Hamming NN:

ابتدا تمام اثر انگشت‌ها به بردارهای دودویی تبدیل شدند (۱ برای پیکسل‌های تیره، ۰ برای پیکسل‌های روشن).

هر اثر انگشت ذخیره شد و مدل برای هر تصویر جدید، فاصله‌ی همینگ را با تصاویر موجود مقایسه کرد.

تصویری که کمترین فاصله‌ی همینگ را داشت، به عنوان نزدیک‌ترین اثر انگشت انتخاب شد.

چالش‌های Hamming NN:

حساسیت زیاد به نویز (اگر تصویر حتی کمی تغییر کند، فاصله‌ی همینگ افزایش می‌یابد)

نیاز به ذخیره‌ی تمام نمونه‌ها، چون یادگیری انجام نمی‌شود

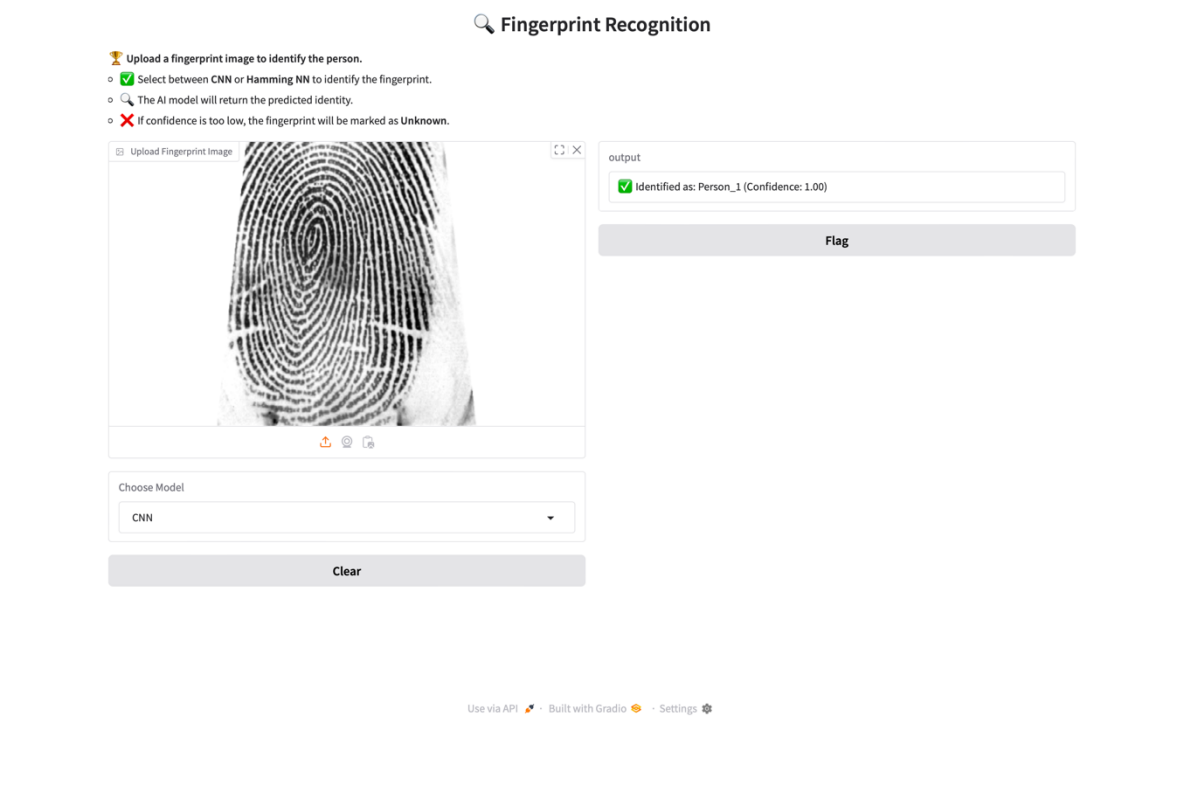
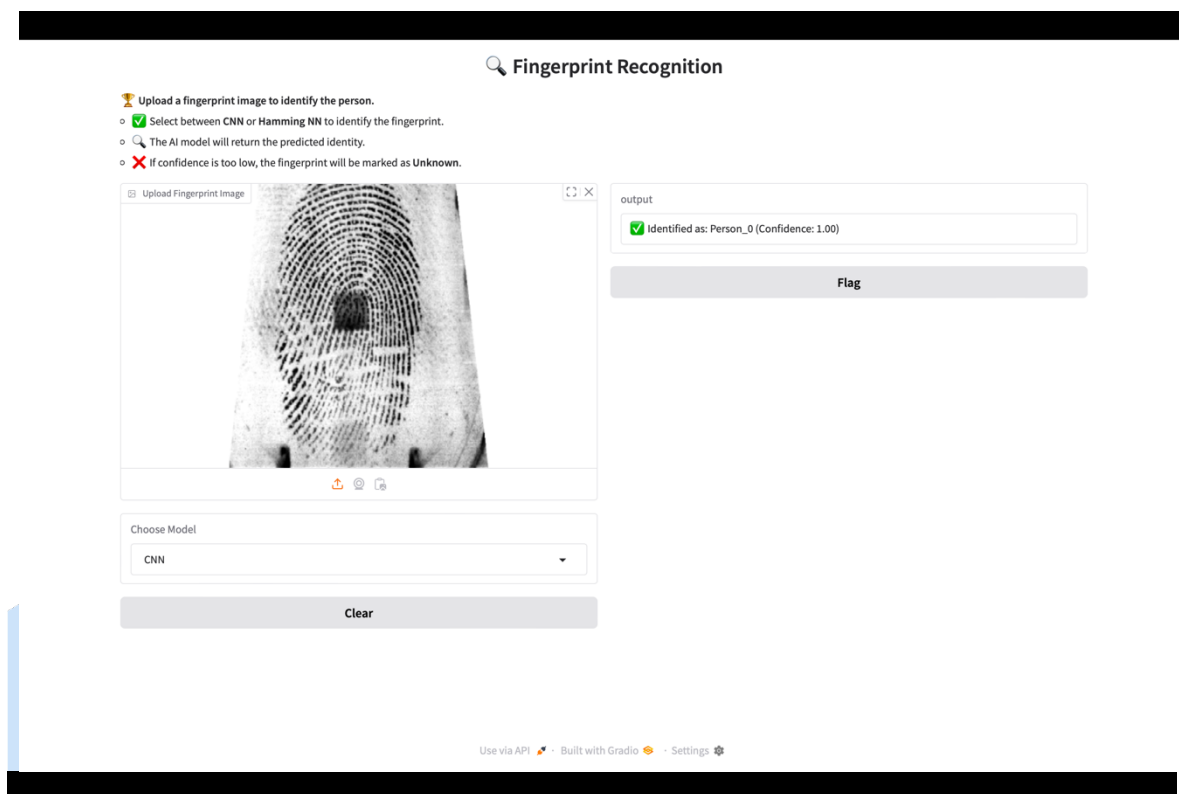
## نتایج و تحلیل آن‌ها

ابتدا ۱۰ اپاک آخر که برای آموزش دیدن مدل نهایی cnn دیده شد را می‌آورم.

```
Epoch 40/50
44/44 ----- 124s 3s/step - accuracy:
0.9900 - loss: 0.0350
Epoch 41/50
44/44 ----- 122s 3s/step - accuracy:
0.9962 - loss: 0.0080
Epoch 42/50
44/44 ----- 124s 3s/step - accuracy:
0.9941 - loss: 0.0192
Epoch 43/50
44/44 ----- 141s 3s/step - accuracy:
0.9968 - loss: 0.0163
Epoch 44/50
44/44 ----- 144s 3s/step - accuracy:
0.9980 - loss: 0.0072
Epoch 45/50
44/44 ----- 121s 3s/step - accuracy:
0.9987 - loss: 0.0039
Epoch 46/50
44/44 ----- 124s 3s/step - accuracy:
0.9968 - loss: 0.0069
Epoch 47/50
44/44 ----- 125s 3s/step - accuracy:
0.9958 - loss: 0.0121
Epoch 48/50
44/44 ----- 122s 3s/step - accuracy:
1.0000 - loss: 0.0027
Epoch 49/50
44/44 ----- 124s 3s/step - accuracy:
0.9999 - loss: 0.0018
Epoch 50/50
44/44 ----- 124s 3s/step - accuracy:
1.0000 - loss: 2.5399e-04
```

همگرا شدن و به سمت صفر رفتن loss نشان‌دهنده‌ی دقت ۱۰۰ درصدی مدل در تشخیص اثر انگشت‌های موجود در دیتاست است. این یعنی لازم نیست آموزش بیش از این ادامه پیدا کند. حالا می‌ماند تست دستی مدل با عکس‌های انتخابی توسط خودمان. این‌جا تصمیم بر این گرفتم که کار را راحت‌تر کنم.

یک ui با استفاده از کتابخانه‌ی gradio در پایتون و گوگل کولب درست کردم که به کار آلود عکس و تست کردن آن را بسیار ساده کرد. در پایین تصاویری را که مدل را با آن ها در محیط gradio تست کردم می‌آورم:




## Fingerprint Recognition

🏆 Upload a fingerprint image to identify the person.

- ✅ Select between CNN or Hamming NN to identify the fingerprint.
- 🔍 The AI model will return the predicted identity.
- ❌ If confidence is too low, the fingerprint will be marked as **Unknown**.

Upload Fingerprint Image



output

✅ Identified as: Person\_2 (Confidence: 1.00)

Flag

Choose Model

CNN

Clear


Use via API 🔗 · Built with Gradio 🍷 · Settings ⚙️

## Fingerprint Recognition

🏆 Upload a fingerprint image to identify the person.

- ✅ Select between CNN or Hamming NN to identify the fingerprint.
- 🔍 The AI model will return the predicted identity.
- ❌ If confidence is too low, the fingerprint will be marked as **Unknown**.

Upload Fingerprint Image



output

✅ Identified as: Person\_3 (Confidence: 1.00)

Flag

Choose Model

CNN

Clear

Use via API 🔗 · Built with Gradio 🍷 · Settings ⚙️

بدین ترتیب محیط و صحت پیش‌بینی‌ها را دیدیم. حالا مدل را روی یک نمونه تصویر نویزی. یک نمونه تصویر بی کیفیت و یک نمونه تصویر خارج از دیتا چک می‌کنم.




**Fingerprint Recognition**

🏆 Upload a fingerprint image to identify the person.

- ✅ Select between CNN or Hamming NN to identify the fingerprint.
- 🔍 The AI model will return the predicted identity.
- ❌ If confidence is too low, the fingerprint will be marked as Unknown.

Upload Fingerprint Image



output

✅ Identified as: Person\_3 (Confidence: 1.00)

Flag

Choose Model

CNN

Clear


Use via API • Built with Gradio • Settings

**Fingerprint Recognition**

🏆 Upload a fingerprint image to identify the person.

- ✅ Select between CNN or Hamming NN to identify the fingerprint.
- 🔍 The AI model will return the predicted identity.
- ❌ If confidence is too low, the fingerprint will be marked as Unknown.

Upload Fingerprint Image



output

✅ Identified as: Person\_3 (Confidence: 1.00)

Flag

Choose Model

CNN

Clear

Use via API • Built with Gradio • Settings


## حال دو نمونه خارج دیتا:

### Fingerprint Recognition

👤 Upload a fingerprint image to identify the person.

- ✅ Select between CNN or Hamming NN to identify the fingerprint.
- 🔍 The AI model will return the predicted identity.
- ❌ If confidence is too low, the fingerprint will be marked as **Unknown**.

Upload Fingerprint Image



Choose Model

CNN

Clear

output

❌ Unknown Fingerprint (Confidence: 1.00)

Flag


[Use via API](#) • [Built with Gradio](#) • [Settings](#)

### Fingerprint Recognition

👤 Upload a fingerprint image to identify the person.

- ✅ Select between CNN or Hamming NN to identify the fingerprint.
- 🔍 The AI model will return the predicted identity.
- ❌ If confidence is too low, the fingerprint will be marked as **Unknown**.

Upload Fingerprint Image



Choose Model

CNN

Clear

output



❌ Unknown Fingerprint (Confidence: 1.00)

Flag




[Use via API](#) • [Built with Gradio](#) • [Settings](#)

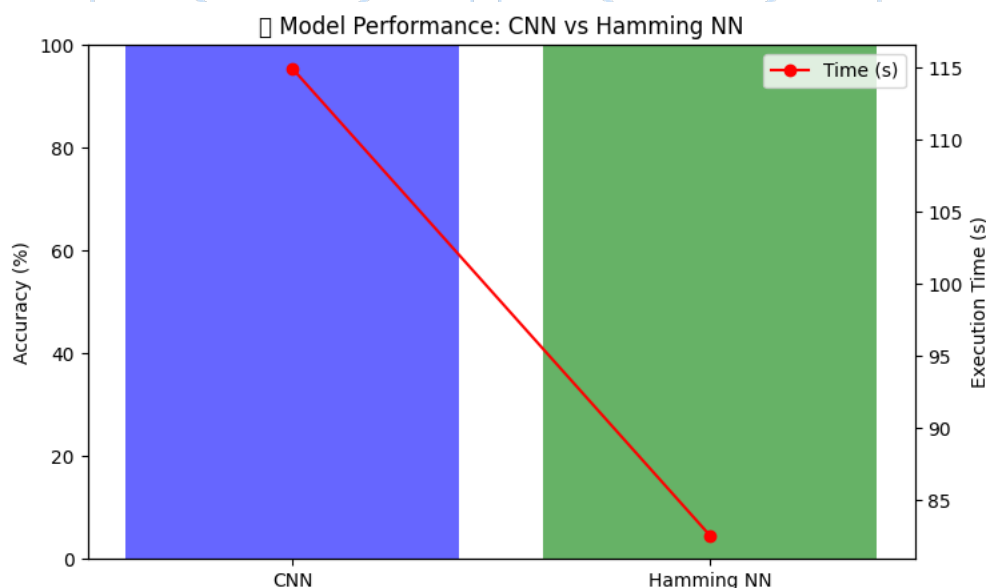
بدین ترتیب عملکرد مطلوب شبکه CNN در شناسایی اثر انگشت‌ها به وضوح مشخص است.  
حال به سراغ دیگر مدل یعنی Hamming NN می‌رویم.

مدل همینگ به سرعت و با دقتی حتی بسیار بالاتر نسبت به مدل قبلی این دیتا را یاد می‌گیرد. این مدل قابلیت این را پیدا می‌کند که هر انگشت هر فرد را جداگانه شناسایی کند چیزی که با CNN موفق به دستیابی به آن نشدم..

 \*\*Final Results:\*\*  
 Hamming NN Accuracy: 100.00%

و در نهایت مقایسه‌ی این دو مدل را در انتهای کد آورده‌ام. برای مقایسه‌ی عادلانه خروجی مدل همینگ که درواقع نام یک یک فایل‌ها بود را تبدیل به کلاس‌های موجود در خروجی شبکه‌ی کانولوشنی کردم. هر دو مدل دقت ۱۰۰ درصد را به ارمغان آوردند اما مدل همینگ سرعت بیشتری در پیش‌بینی داشت. البته باید در نظر داشت که مدل همینگ قابلیت شناسایی کلاس unknown را ندارد همچنین به نویز بسیار حساس است.

 \*\*Final Results:\*\*  
 CNN Accuracy: 100.00% (Time: 114.93s)  
 Hamming NN Accuracy: 100.00% (Time: 82.52s)





همچنین در gradio گزینه‌ی انتخاب کردن مدل دلخواه (همینگ یا cnn) را هم برای راحتی استفاده اضافه کردم. در نهایت می‌توان گفت به علت وجود نداشت دیتای زیاد مدل همینگ عملکرد بسیار بهتر و پرسرعت تری داشته اما به نکات و مزایایی که در مورد cnn گفتم حتما باید دقت کرد. تشخیص کلاس unknown در سیستم‌های تشخیص اثر انگشت بسیار حیاتی‌ست. همچنین حساس نبودن به نویز می‌تواند بسیار حیاتی باشد چون حسگرهای اثر انگشت خطا دارند و مدل نباید آنقدر به نویز حساس باشد.

## نتیجه‌گیری نهایی

در این پروژه، به مقایسه دو مدل مختلف برای شناسایی اثر انگشت پرداخته شد: شبکه عصبی کانولوشنی (CNN) و شبکه عصبی همینگ (Hamming NN). هر کدام از این مدل‌ها با چالش‌ها و ویژگی‌های خاص خود همراه بودند که به نوعی انتخاب بین آن‌ها را وابسته به نیازهای مختلف پروژه می‌کرد.

مدل CNN به دلیل قدرت بالای یادگیری ویژگی‌ها از داده‌ها و توانایی در تعمیم‌دهی به تصاویر جدید، انتخاب مناسبی برای شناسایی دقیق اثر انگشت‌ها بود. اما این مدل برای رسیدن به بهترین عملکرد، به داده‌های کافی و زمان پردازش زیاد نیاز داشت. همچنین، استفاده از Augmentation و کلاس "Unknown" برای مواجهه با اثر انگشت‌های جدید و ناشناخته، مدل را به شدت کارآمدتر ساخت.

از سوی دیگر، Hamming NN با استفاده از مقایسه ساده و سریع الگوهای دودویی، می‌تواند به‌طور سریع اثر انگشت‌های موجود را شناسایی کند. اما این مدل از آن جایی که به یادگیری نیازی ندارد، در مواجهه با نویز و تغییرات کوچک در تصویر عملکرد ضعیف‌تری از خود نشان داد. در نتیجه، اگر سرعت و پردازش کمتر اولویت باشد، Hamming NN انتخاب مناسبی است.

در نهایت، این پروژه نشان داد که هر مدل بسته به نیاز خاص خود، می‌تواند نتایج متفاوتی را به ارمغان بیاورد. در حالی که CNN نیاز به زمان و منابع بیشتری دارد، قدرت تعمیم‌دهی آن و توانایی کار با داده‌های پیچیده، آن را به گزینه‌ای مناسب‌تر برای استفاده در پروژه‌های پیچیده‌تر تبدیل کرده است. از سوی دیگر، Hamming NN با سرعت بالا و ساده‌تر بودن، در شرایط خاص و با داده‌های کمتر مناسب‌تر است.

در مجموع، هدف این پروژه بهبود سیستم‌های شناسایی اثر انگشت و طراحی مدلی بود که هم دقت بالایی داشته باشد و هم در زمان‌های مختلف و داده‌های متفاوت به درستی عمل کند. استفاده از مدل‌های ترکیبی و بیشتر کردن داده‌های ورودی می‌تواند به بهبود عملکرد مدل‌های موجود کمک کند.

- به پایان آمد این دفتر حکایت همچنان باقی‌ست. با تشکر از توجه و زحمات شما.