سیستم تشخیص متن فارسی و تطبیق مشخصات شناسنامه‌ای

پروژه پایان ترم

سحر محمدی

موعد تحویل: ۱۱/۲۲

تاریخ تحویل: ۱۱/۲۲

خلاصه:

در این پروژه، به منظور تطبیق اطلاعات و احراز هویت کاربر، از الگوریتم‌های EasyOCR و Hezar برای تشخیص متن فارسی از تصویر شناسنامه استفاده کردم. با ابتکار به تصویر داده شده، ابتدا محل حاوی اطلاعات مورد نظر (نام، نام خانوادگی، تاریخ تولد و ...) را شناسایی کردم. سپس، با استفاده از این اطلاعات، تصاویر سگمنت شده ایجاد کردم و آن‌ها را به مدل‌های Hezar برای تشخیص متن فارسی ارسال کردم.

نتایج به دست آمده را با دقت بررسی کردم و هر مورد اطلاعات را به صورت جداگانه از تصویر استخراج کردم. در اینجا، اهمیت به دقت و صحت گرفتن اطلاعات برای تطبیق آنها با ورودی کاربر مورد نظر است. برای تجربه کاربری، از ویجت‌های IPyWidgets برای ورودی گرفتن و آپلود تصویر استفاده کردم.

شرح تکنیکال

در این پروژه برنامه ای طراحی کرده ام که در ابتدا ۸ فیلد ورودی برای وارد کردن نام، نام خانوادگی، روز تولد، ماه تولد، سال تولد، نام پدر و نام مادر وجود دارد و همچنین یک قسمت برای آپلود عکس تصویر شناسنامه است. و یک دکمه "اعتبارسنجی"۰ این دکمه بعد از آپلود تصویر نمایان می‌شود.

عکس شماره ۱

کاربر با وارد کردن فیلدها و کلیک روی این دکمه،

اگر یکی از فیلدها خالی باشد، پیام "لطفا همه ورودی‌ها را پر کنید!" نشان داده می‌شود.

و اما اگر مشکلی نباشد، کاربر پیام "اطلاعات شما دریافت شد. لطفا برای اعتبارسنجی صبر کنید." را مشاهده خواهد کرد.

پس از این با استخراج متن‌های موجود در تصویر شناسنامه و مقایسه آنها با متن دریافت شده از کاربر برای هر قسمت، اعتبارسنجی انجام می‌شود. اگر نام وارد شده با نام موجود در شناسنامه برابر باشد، عبارت "نام معتبر است" چاپ می‌شود. در غیر این صورت عبارت "نام اشتباه وارد شده!" چاپ می‌شود. و به همین ترتیب این عمل برای همه فیلدها اجرا می‌شود.

در این پروژه، برای بخش تشخیص متن فارسی از روی تصویر، از مدل از پیش آموزش دیده شده کتابخانه hezar استفاده شده. در ادامه به توضیحات بیشتر در مورد این کتابخانه می‌پردازیم.

Hezar یک کتابخانه چندمنظوره هوش مصنوعی است که برای ساده‌تر کردن توسعه هوش مصنوعی برای جامعه فارسی زبانان طراحی شده است. اینجا چگونگی عملکرد آن را مشاهده می‌کنید:

مجموعه کامل Hezar :بهترین مدل‌ها و ابزارهای هوش مصنوعی را برای وظایف زبان فارسی گردآوری کرده است. اگر در حوزه پردازش زبان طبیعی(NLP) ، بینایی کامپیوتری، صدا یا حالات دیگر کار می‌کنید، Hezar همیشه در دسترس شماست.

یکپارچه‌سازی آسان: استفاده از مدل‌های هوش مصنوعی با Hezar به اندازه نوشتن چند خط کد ساده است. این به طور سریع با Hugging Face Hub یکپارچه می‌شود، که دسترسی آسان به مدل‌های پیش‌آموزش دیده فراهم می‌کند.

رابط کاربری واقعاً مناسب برای توسعه‌دهندگان Hezar :یک رابط کاربری واقعاً مناسب برای توسعه‌دهندگان فراهم می‌کند. شما می‌توانید به سرعت مدل‌ها را بارگیری و برای وظایف مختلف بدون راه‌اندازی پیچیده استفاده کنید.

رابط مبتنی بر وظیفه مدل Hezar :مدل‌ها را بر اساس وظایف (مانند تحلیل احساسات، برچسب‌گذاری بخش‌های کلامی، تشخیص نامداری و ...) سازماندهی می‌کند. این رویکرد برای کاربران عمومی محیط کار را راحت‌تر می‌کند.

ابزارهای اضافی Hezar :علاوه بر مدل‌ها، ابزارهایی نظیر جاسازی واژه‌ها، توکن‌سازها، استخراج ویژگی‌ها و ... را شامل می‌شود.

راه‌اندازی و بنچمارک Hezar :همچنین ابزارهای اضافی برای راه‌اندازی، بنچمارک، بهینه‌سازی و سایر وظایف یادگیری ماشین ارائه می‌دهد.

Hezarاز ترکیب مدل‌های یادگیری عمیق و تکنیک‌های پردازش زبان طبیعی (NLP) برای تشخیص متون فارسی استفاده می‌کند. این کتابخانه شامل یک مجموعه جامع از مدل‌های پیش‌آموزش دیده برای وظایف مختلف مانند تحلیل احساسات، برچسب‌گذاری بخش‌های کلامی، تشخیص نامداری و غیره است. این مدل‌ها بر روی حجم زیادی از داده‌های متنی فارسی آموزش دیده شده‌اند و قادر به تشخیص الگوها در متن برای انجام وظایف خاص هستند.

کتابخانه Hezar با استفاده از زبان Python ساخته شده است.

مجموعه دادگان:

اگرچه جزئیات خاصی از دادگان مورد استفاده توسط Hezar در اطلاعات موجود صراحتاً ذکر نشده است، استفاده از دادگان عمومی و تأمین شده برای آموزش مدل‌ها در کتابخانه‌های هوش مصنوعی عمل معمولی است. این دادگان شامل حوزه‌های مختلفی از جمله متن، تصاویر و صدا می‌شوند.

برای وظایف پردازش زبان طبیعی فارسی، احتمالاً Hezar از دادگانی با متون برچسب‌گذاری شده برای وظایفی نظیر تحلیل احساسات، برچسب‌گذاری بخش‌های کلامی، تشخیص نامداری و غیره استفاده می‌کند. این دادگان به مدل‌ها کمک می‌کنند تا زبان فارسی را به طور مؤثر درک و پردازش کنند.

شبکه‌های یادگیری عمیق:

Hezarاز مدل‌های یادگیری عمیق پیش‌آموزش دیده برای وظایف مختلف استفاده می‌کند. گرچه جزئیات دقیق معماری در اینجا مشخص نشده است، اما می‌توانم برخی دیدگاه‌های عمومی ارائه دهم:

BERT (Bidirectional Encoder Representations from Transformers): BERTیک مدل مبتنی بر ترانسفورمر است که نمایانگرهای کلمات منطقی را با در نظر گرفتن همزمان متن‌های چپ و راست یاد می‌گیرد. به دلیل کارآیی آن، BERTبه عنوان یکی از بهترین‌ها در وظایف پردازش زبان طبیعی مورد استفاده قرار گرفته است.

CNN (Convolutional Neural Networks): شبکه‌های عصبی کانولوشنی معمولاً برای وظایف بینایی کامپیوتری استفاده می‌شوند. آن‌ها از لایه‌های کانولوشنی که ویژگی‌های سلسله‌مراتبی از داده‌های تصویری را یاد می‌گیرند، تشکیل شده‌اند.

LSTMs: LSTM (Long Short-Term Memory)یک نوع شبکه عصبی بازگشتی (RNN) هستند که برای ضبط وابستگی‌های متوالی طراحی شده‌اند. آن‌ها برای وظایفی که دنباله دارند مانند تشخیص گفتار یا مدل‌سازی زبان مفید هستند.

ترانسفورمر: ترانسفورمرها، از جمله BERT ، GPTو دیگران، در زمینه پردازش زبان طبیعی نقشهای اساسی ایفا کرده‌اند. آن‌ها از مکانیزم‌های خودتوجه برای مدل‌سازی سیاق استفاده می‌کنند و نتایجی بهتر در وظایف مختلف زبانی حاصل کرده‌اند.

سایر معماری‌ها: Hezar ممکن است شامل معماری‌های دیگری برای وظایف خاص باشد، مانند CRFs (Conditional Random Fields) برای برچسب‌گذاری توالی یا مدل‌های مبتنی بر توجه برای توضیح تصویر.

البته مشکلی که وجود دارد این مدل فقط جملات کوتاه را به خوبی تشخیص می‌دهد. برای اینکه از یک تصویر بزرگ، متن های کوتاه یک جمله ای را استخراج کنم، از کتابخانه easyocr استفاده کردم که در ادامه بیشتر توضیح می‌دهم.

EasyOCRیک بسته‌ی پایتونی است که یک راه آسان برای استخراج متن از تصاویر فراهم می‌کند. بیایید به جزئیات آن بپردازیم:

نحوه عملکرد EasyOCR :از PyTorch به عنوان دستگاه پشتیبانی از پشته خود استفاده می‌کند. این بسته از تشخیص و تعریف متن برای شناسایی و خواندن متن از تصاویر استفاده می‌کند .این فرآیند شامل دو مرحله اصلی است :تشخیص متن: یافتن مناطق در تصویر که متن در آن وجود دارد EasyOCR .برای این منظور از الگوریتم CRAFTاستفاده می‌کند .تعریف متن: رمزگشایی مناطق متن شناسایی شده به کلمات واقعی. مدل تعریف بر معماری (CRNN (Convolutional Recurrent Neural Network مبتنی بر اجزاءی نظیر استخراج ویژگی) استفاده از ResNet یا(VGG ، برچسب‌گذاری دنباله (LSTM) و رمزگشایی (CTC) است .می‌توانید بین روش‌های مختلف رمزگشایی انتخاب کنید، از جملهgreedy ، beam searchیا word beam search. زبان‌های پشتیبانی شده EasyOCR :به شما این امکان را می‌دهد که متن را در زبان‌های مختلف تشخیص دهید. می‌توانید یک لیست از کدهای زبان) مثلاً'ch_sim'] ، ['en'])را مشخص کنید تا زبان‌هایی که می‌خواهید تشخیص دهید را نشان دهید .این بسته از شتاب‌دهنده GPU پشتیبانی می‌کند (اگر موجود باشد) و گزینه‌هایی برای دایرکتوری‌های ذخیره‌سازی مدل فراهم می‌کند .پارامترها و سفارشی‌سازی :کلاس Reader در EasyOCR چندین پارامتر برای سفارشی‌سازی فراهم می‌کند :lang_list :زبان‌های مورد تشخیص را مشخص کنید :gpu . فعال کردن پشتیبانی از) GPU پیش‌فرض: صحیح :model_storage_directory .(مسیر به دایرکتوری حاوی داده‌های مدل :download_enabled .فعال کردن دانلود مدل در صورت عدم یافتن محلی . :recog_networkانتخاب شبکه تعریف متن خود) به عنوان مثال 'standard' یا یک شبکه سفارشی .(:detectorبارگذاری مدل تشخیص متن :recognizer .بارگذاری مدل تعریف متن .و بیشتر!

الگوریتم (CRAFT (Character Region Awareness for Text Detection یک الگوریتم تشخیص متن است که برای یافتن نواحی حاوی متن در تصاویر استفاده می‌شود. الگوریتم CRAFT از یادگیری عمیق به منظور تشخیص نواحی متن بهره می‌برد. در ادامه، مراحل اصلی الگوریتم CRAFT را مرور خواهیم کرد:

۱.تبدیل تصویر به نگاشت هیتمپ

الگوریتم CRAFT با استفاده از یک شبکه عصبی یادگیری عمیق، تصویر را به یک نگاشت هیتمپ (heat map)تبدیل می‌کند. در این هیتمپ، نواحی حاوی متن با مقادیر بالا مشخص می‌شوند.

۲.تولید نمایش چگالی متن

با استفاده از نگاشت هیتمپ، یک نمایش چگالی متن ایجاد می‌شود. این نمایش نقاطی را که احتمال حاوی متن بالاست، به شکل چگالی نشان می‌دهد.

۳.استفاده از افقی-عمودی گرادیان

CRAFT از گرادیان‌های افقی و عمودی در نمایش چگالی متن برای تشخیص لبه‌ها و مرزهای متن استفاده می‌کند.

۴.استفاده از یادگیری شده برای یافتن نواحی متن

با استفاده از اطلاعات گرادیان و یادگیری شده، CRAFTنواحی حاوی متن را شناسایی می‌کند. این نواحی ممکن است شامل لبه‌ها و مرزهای متن باشند.

۵.پس‌پردازش و ادغام نواحی

پس از شناسایی نواحی حاوی متن، یک فرآیند پس‌پردازش انجام می‌شود تا نواحی متنی ادغام شوند و یک نتیجه نهایی برای تشخیص متن در تصویر به‌دست آید.

الگوریتم CRAFT به خوبی برای تشخیص متن در تصاویر با اشکال و زوایای مختلف عمل می‌کند و از یک ترکیب هوش مصنوعی و تکنیک‌های پردازش تصویر برای دست‌یابی به نتایج دقیق استفاده می‌کند.

ساختار خروجی تابع **reader.readtext(image)**در کتابخانه EasyOCR به شکل یک لیست از تاپل‌ها (tuple)است که هر تاپل اطلاعات یک متن تشخیص‌داده‌شده را نماینده می‌شود. هر تاپل شامل چهار عنصر است:

۱. **موقعیت گوشه‌های مستطیل محدب :(Bounding Box)** این عنصر شامل یک لیست از چهار جفت اعداد است که نقاط گوشه‌های مستطیل محدب (Bounding Box) را در تصویر نشان می‌دهد. هر جفت اعداد به ترتیب **(x, y)**نمایش داده می‌شود.

۲. **متن تشخیص‌داده‌شده :**این عنصر شامل متنی است که از تصویر تشخیص داده شده است و به عنوان یک رشته (string) نمایش داده می‌شود.

۳. **احتمال تشخیص متن :**این عنصر نمایانگر احتمال (probability) تشخیص متن است.

۴. **متن خالی :(Optional)** در برخی موارد، این عنصر می‌تواند متن خالی باشد، یعنی یک رشته خالی یا **None**.

پس ابتدا با توجه به محل های یافته شده توسط easyocr منطقه ای از تصویر که در آن یک جمله وجود دارد را پیدا میکنم و از تصویر اولیه استخراج میکنم. اما چون کیفیت متن تشخیص داده شده توسط این الگوریتم به اندازه کافی دقیق نیست، فقط منطقه را پیدا کردم و سپس هر سگمنت را به کمک مدل هزار تشخیص متن دادم.

به طور کلی مجموع داده های موجود برای زبان فارسی بسیار محدود است. همچنین باید به این نکته توجه کنیم که هر حرف فارسی، بر اساس موقعیت آن در کلمات، ممکن است تا چهار سبک نوشتاری مختلف داشته باشد، به عبارت دیگر، جایگاه آغاز (مانند ض)، جایگاه میانی (مانند ض)، جایگاه پایانی (مانند ض) یا سبک جایگاه تنها (مانند ض).

شرح نتایج

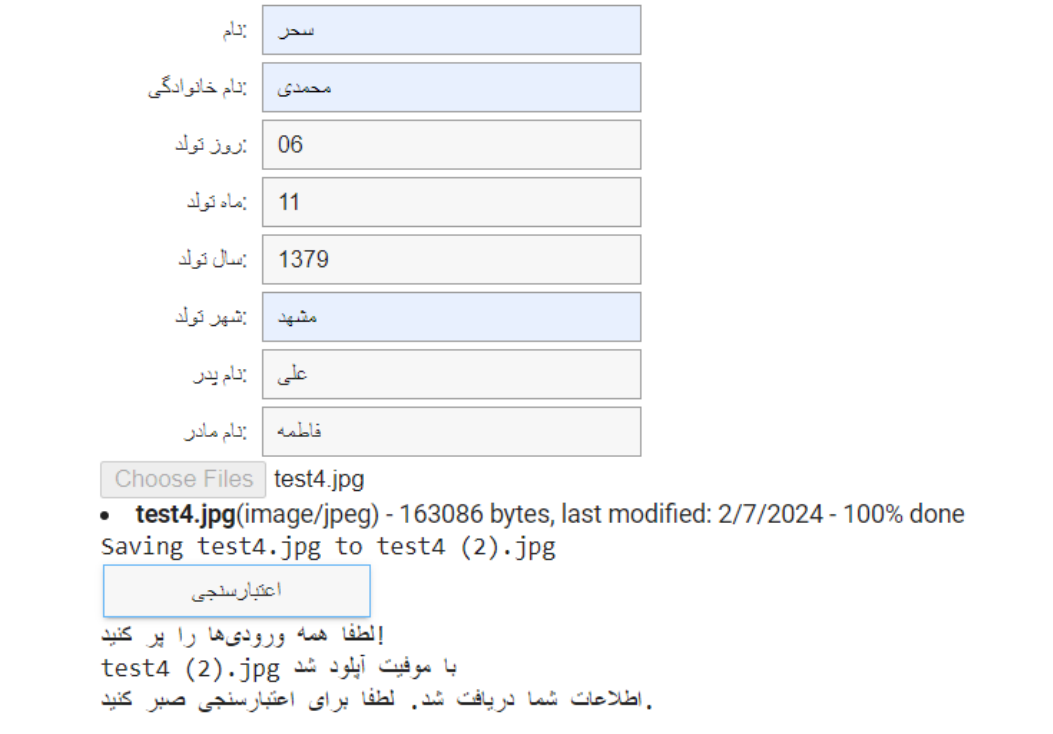
همان طور که گفتیم، به کمک ipywidget ۸ تا فیلد ورودی برای دریافت اطلاعات نام، نام خانوادگی، روز تولد، ماه تولد، سال تولد، نام پدر و نام مادر از کاربر، ایجاد کردم. سپس یک ویجت برای آپلود تصویر اما آن را ابتدا برابر none قرار دادم چون در صورت استفاده از این ویجت تا آپلود تصویر، بقیه کد اجرا نمی‌شود بنا بر این نمیتوان به کمک تابع display ویجت‌های دریافت اطلاعات نام و این موارد. در نهایت هم یک باتن ویجت برای دکمه.

برای دکمه یک تابع تعریف کردم که با کلیک کردن دکمه آن تابع صدا زده شود. در این تابع ابتدا خالی نبودن ویجت تصویر چک میشود. در مرحله بعد خالی نبودن هر یک از ویجت های متنی. سپس تصویر دریافت شده را ذخیره میکند. و اصلاعات متنی را در یک دیکشنری ذخیره میکند. در نهایت یک تابع برای تطبیق اطلاعات صدا زده میشود.

```
import ipywidgets as widgets
from IPython.display import display, clear_output
from google.colab import files

# Define text widgets
name = widgets.Text(description='نام:')
last_name = widgets.Text(description='نام خانوادگی:')
birth_day = widgets.Text(description="روز تولد:")
birth_month = widgets.Text(description="ماه تولد:")
birth_year = widgets.Text(description="سال تولد:")
birth_city = widgets.Text(description="شهر تولد:")
father_name = widgets.Text(description="نام پدر:")
mother_name = widgets.Text(description="نام مادر:")

# Create a container to hold the text widgets
text_widgets_container = widgets.VBox([name, last_name, birth_day,
birth_month, birth_year, birth_city, father_name, mother_name])

# Upload a photo
photo_upload = None

# Create a button widget
button_widget = widgets.Button(description='اعتبارسنجی')

# Define a callback function for the button click event
def on_button_click(b):
    global photo_upload  # Use the global variable to track photo upload
status
```

```
    #Validate the entered text (you can add your custom validation logic
here)
    if not all(widget.value for widget in [name, last_name, birth_day,
birth_month, birth_year, birth_city, father_name, mother_name]):
        print("لطفا همه ورودی‌ها را پر کنید!")
        return

    for filename, content in photo_upload.items():
        with open(filename, 'wb') as f:
            f.write(content)
        print(f"{filename} با موفیت آپلود شد")


    # Access the entered text
    entered_text = {widget.description: widget.value for widget in [name,
last_name, birth_day, birth_month, birth_year, birth_city, father_name,
mother_name]}

    # Print a message
    print("اطلاعات شما دریافت شد. لطفا برای اعتبارسنجی صبر کنید.")

    func(filename, entered_text)

# Attach the callback function to the button click event
button_widget.on_click(on_button_click)

# Display widgets
display(text_widgets_container)

# Enable users to upload a photo
photo_upload = files.upload()

display(button_widget)
```

عکس شماره ۲



عکس شماره ۳

تابع تطبیق که ورودی آن نام تصویر و دیکشنری از متن های وارد شده است، به این صورت است که همان طور که در شرح تکنیکال اشاره شد، با استفاده از easyocr محل متن‌های موجود در تصویر شناسنامه تشخیص داده میشود به صورت ۴ نقطه که مستطیلی را مشخص میکند که در ان یک متن کوتاه وجود دارد. این محل از تصویر اصلی استخراج شده و به صورت یک تصویر جدید ذخیره میشود.

در مرحله بعد این تصاویر یا سگمنت های ذخیره شده یکی لود میشوند و به کمک دو مدل موجود در کتابخانه hezar متن موجود در آن شناسایی میشود. از هر دو مدل استفاده کردم چون برای برخی متن ها مدل TrOCR خوب جواب میدهد و در برخی متن های دیر مدل CRNN به درستی جواب میدهد. همه متن های استخراج شده در یک ارایه all_text ذخیره میشود.

در نهایت، تطبیق انجام میشود به این گونه که به عنوان مثال برای نام بررسی میشود که آیا نام در متن های استخراج شده در تصویر موجود است یا نه؟ اگر بله عبارت نام معتبر است. چاپ میشود. در غیر این صورت عبارت نام اشتباه وارد شده.

برای نام خانوادگی، نام مادر، نام پدر و شهر تولد به همین صورت انجام میشود. برای تاریخ تولد به این صورت است که در یک متغیر ابتدا سال تولد، بعد کاراکتر "۱" را به کنارش اضافه میکنم، بعد ماه تولد، دوباره کاراکتر "۱" و سپس روز تولد. ۱ های اضافه شده به این خاطر است که کاراکتر "/" توسط مدل تشخیص استفاده شده "۱" تشخیص داده میشود. نکته دیگری که وجود دارد این است که اعداد وارد شده در برنامه انگلیسی هستند و در متغیر هم به صورت انگلسی ذخیره شده اند. برای همین با استفاده از کتابخانه convert_numbers اعداد انگلیسی را به فارسی تبدیل میکنم. در نهایت بررسی میکنم که این عبارت در متن های استخراج شده در تصویر وجود دارد یا خیر.

```
import easyocr
import cv2
import os
import numpy as np
from PIL import Image
import convert_numbers
from hezar.models import Model

def ocr_and_save_segments(image_path, output_folder, language='fa'):
    # Create the output folder if it doesn't exist
    if not os.path.exists(output_folder):
        os.makedirs(output_folder)

    # Load the image
```

```
    image = cv2.imread(image_path)

    # Initialize the EasyOCR reader with the specified language
    reader = easyocr.Reader([language])

    # Perform OCR on the image
    result = reader.readtext(image)

    # Process each detected segment
    for i, detection in enumerate(result):
        points = detection[0]
        box = [[int(x), int(y)] for (x, y) in points]
        box = np.array(box)

        # Extract the region of interest (ROI) from the image
        x, y, w, h = cv2.boundingRect(box)
        segment = image[y:y+h, x:x+w]

        # Save the segment to the output folder
        segment_filename = f'segment_{i + 1}.png'
        segment_path = os.path.join(output_folder, segment_filename)
        cv2.imwrite(segment_path, segment)

def ocr_with_hezar(image_path, model):
    # Load the image
    image = Image.open(image_path)

    # OCR with CRNN
    texts = model.predict(image)
    return texts[0]['text']

def check(target_text, all_texts):
  for text in all_texts:
      if target_text == text:
          return 1
  return 0

def func(file_name, entered_text):
    # Example usage for Farsi (Persian)
    image_path = file_name  # Replace with the path to your Farsi text
image
    output_folder = '/content/output_segments'
    ocr_and_save_segments(image_path, output_folder, language='fa')

    # Example usage for Farsi (Persian)
```

```
    image_folder = output_folder

    # Process each segmented image in the folder
    for filename in os.listdir(image_folder):
        if filename.endswith(".png"):
            image_path = os.path.join(image_folder, filename)

    # لیستی برای ذخیره تمام متن‌ها
    all_texts = []

    # OCR with TrOCR
    model1 = Model.load("hezarai/trocr-base-fa-v2")

    # OCR with CRNN
    model2 = Model.load("hezarai/crnn-fa-printed-96-long")

    # Process each segmented image in the folder
    for filename in os.listdir(image_folder):
        if filename.endswith(".png"):
            image_path = os.path.join(image_folder, filename)

            # Run OCR with hezar on the segmented image
            text = ocr_with_hezar(image_path, model1)
            all_texts.append(text)

            text = ocr_with_hezar(image_path, model2)
            all_texts.append(text)


    if(check(entered_text['نام:'], all_texts)):
        print("نام معتبر است")
    else:
        print("نام اشتباه وارد شده!")
    if(check(entered_text['نام خانوادگی:'], all_texts)):
        print("نام خانوادگی معتبر است")
    else:
        print("نام خانوادگی اشتباه وارد شده!")
    if(check(entered_text['نام مادر:'], all_texts)):
        print("نام مادر معتبر است")
    else:
        print("نام مادر اشتباه وارد شده!")
    if(check(entered_text['نام پدر:'], all_texts)):
        print("نام پدر معتبر است")
    else:
        print("نام پدر اشتباه وارد شده!")
```

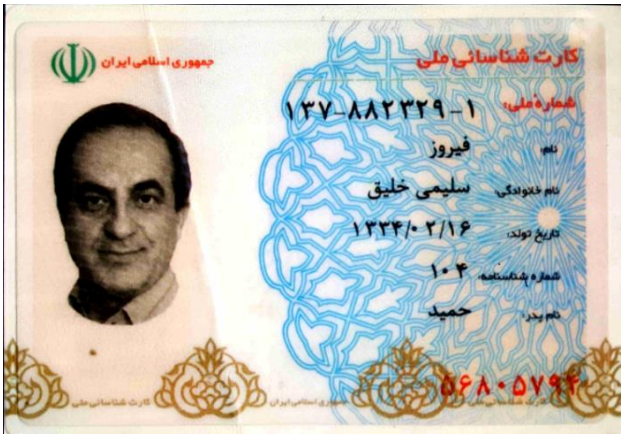
```
    if(check(entered_text['شهر تولد:'], all_texts)):
        print("شهر تولد معتبر است")
    else:
        print("شهر تولد اشتباه وارد شده!")

    birth = entered_text['سال تولد:']+'1'+entered_text[' ماه
تولد:']+'1'+entered_text['روز تولد:']
    birth = convert_numbers.english_to_persian(birth)

    if(check(birth, all_texts)):
        print("تاریخ تولد معتبر است")
    else:
        print("تاریخ تولد اشتباه وارد شده!")
```

نام معتبر است
نام خانوادگی معتبر است
!نام مادر اشتباه وارد شده
!نام پدر اشتباه وارد شده
شهر تولد معتبر است
تاریخ تولد معتبر است

تصویر شماره ۴

تصویر شماره ۵

با توجه به تصویر شناسنامه میبینیم که نام مادر بی بی فاطمه بوده اما بدلیل اینکه فاطمه وارد شده، عبارت نام مادر اشتباه وارد شده در خروجی نمایش داده شده است. همچنین نام پدر علیرضا است اما بدلیل علی وارد شدن، عبارت نام پدر اشتباه وارد شده مشاهده میشود و اما بقیه موارد که به درسی وارد شده اند عبارت معتبر است را میبینیم.



نام: فیروز

نام خانوادگی: سلیمی خلیق

روز تولد: 16

ماه تولد: 02

سال تولد: 1334

شهر تولد: مشهد

نام پدر: حمید

نام مادر: مریم

Choose Files tmp95BB.jpg

- **tmp95BB.jpg**(image/jpeg) - 427824 bytes, last modified: 11/20/2023 - 100% done

```
Saving tmp95BB.jpg to tmp95BB.jpg
```

اعتبارسنجی

```
لطفا همه ورودی‌ها را پر کنید!
WARNING:easyocr.easyocr:Neither CUDA nor MPS are available - defaulting to CPU. N
tmp95BB.jpg با موفیت آپلود شد
اطلاعات شما دریافت شد. لطفا برای اعتبارسنجی صبر کنید.
نام معتبر است
نام خانوادگی معتبر است
نام مادر اشتباه وارد شده!
نام پدر اشتباه وارد شده!
شهر تولد اشتباه وارد شده!
تاریخ تولد معتبر است
```

نام و نام خانوادگی و تاریخ تولد به درستی اعتبارسنجی شده اند. نام مادر و شهر تولد در تصویر وجود نداشت پس منطقی است که نتوانسته تطبیقی برای ان پیدا کند اما در مورد نام پدر ضعف سیستم باعث عدم تشخیص است.

نام: مجتبی

نام خانوادگی: الیاسی

روز تولد: 20

ماه تولد: 06

سال تولد: 1363

شهر تولد: مشهد

نام پدر: صفا

نام مادر: سحر

Choose Files tmpCBD22.jpg

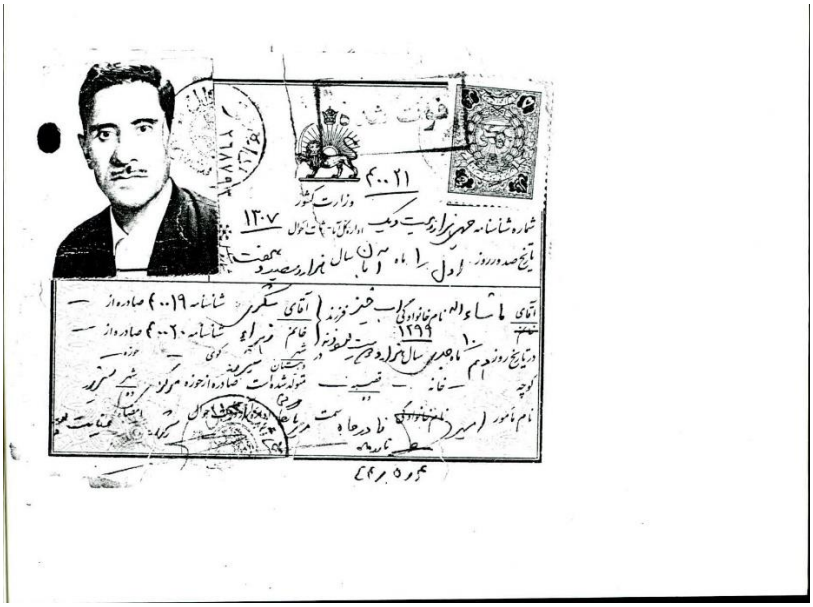
- **tmpCBD22.jpg**(image/jpeg) - 43475 bytes, last modified: 2/13/2024 - 100% done

```
Saving tmpCBD22.jpg to tmpCBD22 (5).jpg
```

اعتبارسنجی

```
WARNING:easyocr.easyocr:Neither CUDA nor MPS are available - defaulting to CP
tmpCBD22 (5).jpg با موفیت آپلود شد
اطلاعات شما دریافت شد. لطفا برای اعتبارسنجی صبر کنید.
نام معتبر است
نام خانوادگی معتبر است
نام مادر اشتباه وارد شده!
نام پدر اشتباه وارد شده!
شهر تولد اشتباه وارد شده!
تاریخ تولد معتبر است
```

اینجا هم به غیر از نام پدر بقیه موارد که در تصویر بوده اند به درستی تطبیق داده شده اند.



نام: ماشاءاله

نام خانوادگی: آب خیز

روز تولد: سیذ

ماه تولد: سیذ

سال تولد: ذش

شهر تولد: سری

نام پدر: شکری

نام مادر: شر

Choose Files tmp2182.jpg

- **tmp2182.jpg**(image/jpeg) - 325880 bytes, last modified: 2/13/2024 - 100% don

```
Saving tmp2182.jpg to tmp2182.jpg
```

اعتبارسنجی

```
WARNING:easyocr.easyocr:Neither CUDA nor MPS are available - defau
tmp2182.jpg با موفیت آپلود شد
اطلاعات شما دریافت شد. لطفا برای اعتبارسنجی صبر کنید.
!نام اشتباه وارد شده
نام خانوادگی معتبر است
!نام مادر اشتباه وارد شده
!نام پدر اشتباه وارد شده
!شهر تولد اشتباه وارد شده
!تاریخ تولد اشتباه وارد شده
```

در اینجا از بین نام، نام خانوادگی و نام پدر که به درستی در ورودی وارد شده اند. فقط نام خانوادگی با وجود اینکه در تصویر به صورت دستنویس است تشخیص داده شده است.

بقیه موارد را قادر به تشخیص نبود.