Language Related Research
E-ISSN: 2383-0816
https://lrr.modares.ac.ir
https://doi.org/10.29252/LRR.14.2.11

Tarbiat Modares University

Vol. 14, No. 2, Tome 74
pp. 363-400
May & June 2023

Correction and Improvement of the Common Processes in Optical Character Recognition (OCR) of Persian Texts: Using the Features of the Persian Script and a Dimension Transference Algorithm

Arash Zareian[1], *Tayebeh Mosavi Miangah*[2*], *Belghis Rovshan*[3] & *Seyed Mostafa Fakhr Ahmad*[4]

Received: 26 June 2021
Received in revised form: 3 December 2021
Accepted: 24 December 2021

Abstract

Since the technology of optical recognition of characters is essentially based on Latin script, almost all the algorithms and processes involved in Persian OCR systems are constructed upon the structure and scriptological features of Latin alphabet. This utilization of the means and features of Latin script to design Persian-based OCR systems however, not only has not resulted in the appropriate optical recognition of Persian characters but it also has simultaneously ended in confusion on the part of both the Persian-speaking users and the systems. This paper, therefore, begins with a short review of the significance of language and linguistics in the field of information technology in connection with OCR systems. Then, it will continue with a short history of Persian/Arabic script, while focusing on the scribal features of Persian writing system and its differences with other scripts. In the next part, for effective utilization of the formal elements of the Persian script, these elements have been categorized according to their application and significance in the process of the user's interaction with Persian OCR systems. Furthermore, through a step by step discussion and analysis of the processes involved in optical recognition of characters based on the scriptological features of the Persian script, not only the deficiencies and faults of the current Latin-based OCR systems will be pinpointed but also a different aspect of the Persian writing system, in connection with its use in computer software, especially OCR systems, will be used so that the reader will practically notice the potentials and capabilities of this complex script in contrast to the simpler Latin writing system. In the end, in order to upgrade and improve the current algorithms employed in Persian OCR systems, the geometrical process of transferring bi-dimensional specifications into mono-dimensional ones has been utilized. The proposed algorithm, which is based on the scriptological features of Persian script, will simultaneously result in the convenient manipulation of patterns, reduction of the bulk of the database, and acceleration of the data processing rate.

Keywords: Optical character recognition, OCR, Dimension Transference Algorithm, Persian writing system, Persian scriptological features

1.PhD. Candidate in Linguistics, Payame Noor University, P.O. Box 19395-3697, Tehran, Iran

2. Corresponding author: Associate Professor in Linguistics, Payame Noor University, P.O. Box 19395-3697, Tehran, Iran, *Email*: *mosavit@pnu.ac.ir*

3. Professor in Linguistics, Payame Noor University, P.O. Box 19395-3697, Tehran, Iran 4. Associate Professor in Computer, Shiraz University, Shiraz, Iran

1. Introduction

Since the technology of optical recognition of characters is essentially based on Latin script, almost all the algorithms and processes involved in Persian OCR systems are constructed upon the structure and scriptological features of Latin alphabet. This utilization of the means and features of Latin script to design Persian-based OCR systems however, not only has not resulted in the appropriate optical recognition of Persian characters but it also has simultaneously ended in confusion on the part of both the Persian-speaking users and the systems. Therefore, in order to present a different portrait of Persian writing system when working with computers, especially in OCR systems, this research, attempts to describe and analyze the processes involved in optical character recognition based on the scriptological features of the Persian alphabet and elaborate on its differences with the existing Latin-based systems. In line with this objective, after reviewing the history and evolution of the Persian script through different periods, this research gives a classified illustration of the scriptological features of the Persian writing system and its formal elements with a special focus on the OCR processes. Consequently, in this study, the formal elements of the Persian script are categorized according to their application and significance in the interaction of the user with the Persian OCR softwares. Furthermore, the effective utilization of these scriptological elements is expressed in the framework of the existing algorithms, as well as, in the form of a proposed algorithm. The proposed algorithm, on the one hand, results in the practical elimination of the high affectation of the existing algorithms when facing the cursiveness and elongation, of the Persian letters, which previously increased the error rate of the OCR processes; and on the other hand, it highly prevents an increase in the bulk of the database and computations, related to the stored patterns, which previously caused a decrease in the software performance.

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

[DOR: 20.1001.1.23223081.1401.0.0.114.6]

[DOI: 10.29252/LRR.14.2.11]

2. Literature Review

The study of Persian/Arabic characters' representation have been studied since 1970s (Bonyani & Jahangard, 2020) and the very beginning algorithms for representing Arabic scripts have been released in 1990s. (Margner & El-Abed, 2008). Many researchers including Shafii (2014) gave up holistic segmentation of Persian characters because of difficulties resulted from some special features of Persian alphabet and only worked on sub words' representation instead. The proposed algorithm of Kiaei (2019), regardless of working on printed limited Omni-font texts did not lead to an accepted results and was inefficient facing to words sequence. Rhmati, et al. (2020) as the latest research in the field of character segmentation like many other studies considered baseline connector as a part of a character and their algorithm suggested a procedure to shorten over length baseline connectors in order to facilitate character recognition through the existing systems. The newly done studies on optical character recognition avoid the structural features in the process of recognition and primarily utilize holistic algorithm based on neural networks inorder to extract distinctive features of characters (Bonyani & Jahangard, 2020).

3. Discussion

Using the concept of baseline connector (BC) in the design of the proposed algorithm, the connected characters will all have an identical BC component. This means that each instance of the BC, regardless of its length, will be identified as one identical component. This way, the BC component of each character and its variable extra stretches are removed by means of algorithms and mathematical processes and replaced by an identical special code. This is different from the common known methods of character segmentation in which the whole character including the BC component goes through an image processing stage. Here, in the pattern comparison stage, the system at first recognizes the BC component and removes its extra stretches and then

compares the remaining letter image with the stored patterns. By removing the BC component from the text image and replacing it with a simple code, contrary to what is customary: 1) the letter segmentation process occurs naturally and successfully; 2) instead of comparing a letter image with all existing patterns, due to the presence of a BC component code, the comparison and recognition process occurs only between the letter image (raw letter) and the patterns belonging to the same set since based on the position of the BC component, the letters can be divided into four sets: a) letter + BC (= initial letters); b) BC + letter (= final letters); c) BC+letter+BC (= medial letters); d) isolated form without a BC on either side (= isolated letters). As a result, instead of comparing the raw letter and trying to match it with all the existing patterns, this comparison is made only between the raw letter (letter image) and the patterns in one of the above four main groups; 3) by removing the BC component, which occurs in various lengths and in practice has no effect on the reading of the word, this component is removed from the letter image and thereby a great number of patterns whose difference lies in their length of the BC component will be eliminated and thus the process of pattern recognition is sped up; 4) on the other hand, the BC component and its extension, i.e. the baseline, divides the letter components into two groups: a) above the baseline; b) below the baseline. The classification of components into upper and lower sets based on the BC component results in further simplification of the pattern comparison process since: a) this way, upper elements are compared only with upper patterns and lower elements are compared only with lower patterns; b) instead of the overall comparison of the raw image frame with the patterns, first the baseline of the raw element is matched with the baseline of the pattern and then the comparison is made in the two upper and lower sections.

4. Conclusion

The functioning status of present Persian OCR soft wares indicates that there are two main challenges in doing research in this field, one related to solving

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

Persian script problems and another concerned with algorithm design and programming. In this study it was determined that the origin of the current challenges is that the programmers ignored the original function and existing philosophy of baseline connector and consider it as a part of the Persian words. This study tried to improve Persian OCR sub word segmentation throughout utilizing an outstanding feature of baseline connector and at the same time its formal elimination. Base line connector forms a large part of Persian texts and its formal deletion from the raw patterns has caused an impressive reduction in the volume of errors in the processing level. Furthermore, considering baseline connector as a criterion can lead to the possibility of Persian scripts and patterns classification. Consequently, instead of comparing one raw element with all patterns, the comparing procedure has limited to homogenous groups and the processing speed has increased. Finally, in order to upgrade and improve the current algorithms employed in Persian OCR systems, the geometrical process of transferring bi-dimensional specifications into mono-dimensional ones has been utilized. The proposed algorithm, which is based on the script logical features of Persian script, will simultaneously result in the convenient manipulation of patterns, reduction of the bulk of the database, and acceleration of the data processing rate.

[DOR: 20.1001.1.23223081.1401.0.0.114.6]

[DOI: 10.29252/LRR.14.2.11]

دوماهنامهٔ بین‌المللی
د۱٤، ش۲ (پیاپی۷٤)، خرداد و تیر ۱٤۰۲، صص ۳٦۳- ٤۰۰
مقاله پژوهشی
http://dorl.net/dor/20.1001.1.23223081.1401.0.0.114.6

ارتقا و اصلاح فرایندهای رایج در بازشناسی نوری حروف متون فارسی با به‌کارگیری ویژگی‌های خط فارسی و الگوریتم انتقال فضا

آرش زارعیان[۱]، طیبه موسوی میانگاه[۲*]، بلقیس روشن[۳]، سید مصطفی فخر احمد[۴]

۱ دانشجوی دکتری گروه زبان‌شناسی، دانشگاه پیام نور، تهران، ایران
۲ دانشیار گروه زبان‌شناسی، دانشگاه پیام نور، تهران، ایران
۳ استاد گروه زبان‌شناسی، دانشگاه پیام نور، تهران، ایران
٤ دانشیار گروه کامپیوتر، دانشکده مهندسی برق و کامپیوتر، دانشگاه شیراز، شیراز، ایران

تاریخ دریافت: ۱٤۰۰/۰٤/۰۵ تاریخ پذیرش:۱٤۰۰/۱۰/۰۳

چکیده

از آنجا که فناوری بازشناسی نوری حروف (اُسی.آر)، اصالتاً برپایهٔ ویژگی‌های خطی لاتین بنا شده است، تقریباً تمام الگوریتم‌ها و مراحل مورد استفاده در نظام‌های رایج بازشناسی حروف فارسی نیز براساس همان ساختار و ویژگی‌های خطوط لاتین گسترش یافته‌اند؛ حال آن‌که، به‌کارگیری ابزار و ویژگی‌های خطوط لاتین در طراحی نظام‌های فارسی‌محور، نه‌تنها درنهایت به انجام بازشناسی صحیح حروف فارسی منجر نشده است، بلکه باعث سردرگمی هم‌زمان نرم‌افزار و کاربر فارسی‌زبان نیز شده است. ازاین‌رو، در اینجا، پس از مقدمه‌ای کوتاه دربارهٔ اهمیت خط و زبان در حوزهٔ فناوری اطلاعات و مروری کلی بر مفهوم نظام‌های بازشناسی نوری حروف و روند تحقیقات مربوطه در ارتباط با خط فارسی و مشکلات موجود در این مسیر، به‌طور خلاصه به سیر تحول خط فارسی در دوره‌های مختلف و شرح ویژگی‌های این خط و تفاوت‌های آن با خطوط دیگر پرداخته شده است. همچنین در ادامه، عناصر شکلی این خط، با توجه به کاربرد و اهمیت آن‌ها در تعامل کاربر با نرم‌افزارهای بازشناسی نوری متون فارسی، طبقه‌بندی و طرز استفادهٔ مؤثر از این عناصر نیز بیان شده است. در این بخش، با توصیف و تحلیل مراحل بازشناسی حروف براساس ویژگی‌های خط فارسی و شرح تفاوت‌های آن با گونه‌های لاتین‌محور موجود، چهره‌ای متفاوت از دستگاه خط فارسی به هنگام کار با رایانه‌ها و به‌ویژه در سیستم‌های بازشناسی نوری حروف عرضه می‌شود، به‌طوری که مخاطب عملاً قابلیت و ظرفیت‌های دستگاه خط فارسی در هماوردی با دستگاه سادهٔ

*نویسنده مسئول: Email: mosavit@pnu.ac.ir

خط لاتین را مشاهده خواهد کرد. در پایان، با اتکا به همین ویژگی‌ها، در جهت ارتقا و اصلاح الگوریتم‌های رایج در بازشناسی نوری حروف فارسی، تسهیل به‌کارگیری الگوها، و تعدیل حجم پایگاه داده‌ها، از فرایند انتقال هندسی فضای دو بُعدی به تک بُعدی نیز بهره جسته‌ایم.

واژه‌های کلیدی: بازشناسی نوری حروف، اُ.سی.آر، الگوریتم انتقال فضا نظام، نگارشی زبان فارسی، ویژگی‌های خطی فارسی.

۱. مقدمه

پیشرفت و گسترش همه‌جانبهٔ فناوری‌های اطلاعاتی و ارتباطی و رشد حیرت‌انگیز و لحظه‌ای آن‌ها، نمایانگر ورود بشر به دوران جدیدی از تحولات جهانی است که از آن به‌عنوان انقلاب ارتباطات یاد می‌کنیم. بی‌شک دور از انتظار نیست که در این دوران اغلب پدیده‌های علمی و فرهنگی از شکل سنتی خود فاصله بگیرند و با هیبتی جدید در این فضای رایانه‌ای و یا به قولی رایاسپهری ظاهر شوند.

در این میان نقش زبان و قالب دیداری آن یعنی خط، در رایاسپهر، به‌عنوان بارزترین و پرکاربردترین ابزار بیان و انتقال مفاهیم و پدیده‌های علمی و فرهنگی، روزبه‌روز برجسته‌تر می‌شود و با توجه به سرعت سرسام‌آور ماشین نوگرایی، همواره این نگرانی وجود دارد که یک زبان خاص به‌واسطهٔ قدرت علمی، اقتصادی و صنعتی کشورهایی که بدان تکلم می‌کنند بیش از همه قدرتمند شود و بر زبان‌های دیگر برتری یابد و درنتیجه فرهنگ‌های دیگر در زیر نفوذ فرهنگی آن زبان محو شوند (عاصی، ۱۳۸۵، ص. ٦٥).

البته امروزه با ارائهٔ راه‌حل‌های گوناگون و ایجاد امکانات مختلف برای همهٔ زبان‌ها، هر گروهی از افراد با هر زبان و فرهنگ این امکان را دارند تا با توسل به فرایند بومی‌سازی، هویت خود را واضح‌تر از همیشه و در زمانی کوتاه‌تر از هر زمان دیگر و در مقیاسی بسیار فراتر از آنچه می‌پنداشتند به ساکنان دهکدهٔ جهانی اعلام کنند. اما برای حضور در رایاسپهر با حفظ هویت ملی و فرهنگی صد البته شرایطی لازم است که مهم‌ترین آن‌ها داشتن سخنی برای گفتن و نیز فراهم آوردن ابزار بیان آن است. اینجاست که بیشترین نمود این رویکرد در حوزهٔ خط و زبان آشکار می‌شود. ازاین‌رو، این مقاله بر آن است تا با شرح و بررسی ویژگی‌های نظام نگارشی

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

فارسی و ارائهٔ راهکارهایی برای به‌کارگیری بهینهٔ این ویژگی‌ها در نظام‌های بازشناسی نوری حروف فارسی، به پژوهشگران و برنامه‌نویسان برای تحقق امر بومی‌سازی و پیش‌روی در رایاسپهر یاری برساند.

۲. مفهوم بازشناسی نوری حروف (OCR[1])

فرایند بازشناسی نوری نویسه‌ها به مجموعه روش‌های تشخیص نواحی متنی در تصاویر اسکن‌شده و تبدیل آن‌ها به متن قابل ویرایش و یا فرایند تبدیل تصویر یک سند به یک متن قابل ویرایش گفته می‌شود (Aranian et al., 2017). امروزه، در هر لحظه حجم بالایی از اسناد کاغذی توسط اسکنرها و یا دوربین‌ها به اسناد تصویری دیجیتالی تبدیل و ذخیره می‌شوند. ذخیره، بازیابی و مدیریت کارآمد این مجموعه‌های تصویری ازجمله نیازها و ضروریات عصر ارتباطات است. با بهره‌گیری از امکانات و قابلیت‌های نظام‌های اُ.سی.آر، امکان استفاده از رایانه در پردازش سریع حجم وسیعی از داده‌های مکتوب در بسیاری از نهادها، سازمان‌ها و مؤسسات فراهم می‌شود. به‌طور عمده استفاده از نظام‌های اُ.سی.آر به یک الزام قطعی بدل شده و علاوه‌بر حذف عامل انسانی در خواندن، جست‌وجو و تایپ مجدد متون، دارای دو مزیت مهم است: اول، سرعت بالای دسترسی به اطلاعات، چراکه برخلاف تصویر در متن امکان ویرایش و جست‌وجو وجود دارد؛ و دوم، کاهش قابل‌ملاحظهٔ فضای ذخیره‌سازی، زیرا حجم فایل‌های متنی بسیار کم‌تر از فایل‌های تصویری است و فایل متنی استخراج‌شده از یک تصویر بسیار کم‌حجم‌تر از خود فایل تصویری است (Hamad et al., 2016).

۳. تاریخچهٔ تحقیق دربارهٔ سیستم‌های OCR فارسی

گفتنی است که مطالعه و کار بر روی بازشناسی نویسه‌های فارسی در دههٔ ۱۹۷۰ م آغاز شد (Bonyani & Jahangard, 2020). اما این امر به‌علت سهل‌انگاری و عدم حمایت مالی از محققان ایرانی، از همان ابتدا به زیرمجموعه‌ای از تحقیقات پیرامون خط عربی بدل شده است، به هر حال، اولین برنامه برای بازشناسی خط عربی، با همهٔ کاستی‌ها و محدودیت‌های آن، در دههٔ ۱۹۹۰ در

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

دسترس کاربران قرار گرفت (Margner & El-Abed, 2008). از آن زمان تاکنون، پژوهشگران داخلی و خارجی با اقتباس و به‌کارگیری شیوه‌های خارجی لاتین‌محور و ویرایش و تعدیل آن‌ها به‌طور مداوم به ابداع شیوه‌های جدید و سازگار با متون فارسی و عربی مشغول بوده‌اند و حجم این تحقیقات، به‌علت اهمیت ویژهٔ امر بازشناسی حروف فارسی/ عربی به‌منظور ذخیرهٔ دیجیتالی انبوهی از متون فارسی و عربی، چنان زیاد است که از ظرفیت این مقاله خارج بوده و پژوهشی جداگانه را می‌طلبد.

از طرفی اغلب پژوهشگران این حوزه، میزان بازشناسی متون در شیوه‌های پیشنهادی خود را نزدیک به ۱۰۰ درصد اعلام می‌کنند که شائبهٔ حل شدن مشکلات بازشناسی حروف فارسی را ایجاد می‌کند. اما باید توجه داشت که این درصدها نسبی بوده و صرفاً به پایگاه داده‌های مورد استفادهٔ آنان محدود است. درنتیجه، به‌علت در دسترس نبودن یک پایگاه جامع داده‌ها که شامل همهٔ انواع متون گرافیکی فارسی‌محور و همهٔ قلم‌های دیجیتالی، چاپی، و دستنویس باشد، میزان حقیقی موفقیت و کارآمدی شیوه‌های موجود عملاً نامشخص است. از آنجا که کانون توجه این مقاله، بر اهمیت خط واصل و نحوهٔ صحیح به‌کارگیری آن در فرایند جداسازی حروف است، در این بخش اشارهٔ مختصری به پژوهش‌های مهم در این زمینه ضروری به‌نظر می‌رسد.

شاید اولین نگاه اصولی به مقولهٔ بازشناسی حروف فارسی را بتوان به پرهامی[۲] و ترقی[۳] (1981) نسبت داد، چراکه آن‌ها برای اولین‌بار با توجه به ماهیت متفاوت خط فارسی، به تجزیهٔ زیرحروف‌ها پرداختند و حروف را براساس بیست ویژگی هندسی طبقه‌بندی کردند. عزمی[۴] و کبیر[۵] (2001) برای جداسازی حروف، الگوریتمی مبتنی بر برچسب‌گذاری مشروط کنتور فوقانی حروف ارائه می‌کنند که برخلاف شیوه‌های قبلی به هم‌پوشانی حروف و قلم‌های مورب حساسیت ندارند. الگوریتم پیشنهادی منهاج[۶] و ادب[۷] (2002) مراحل جداسازی و بازشناسی حروف را با به‌کارگیری شبکه‌های عصبی و توصیفگرهای فوریه با هم ترکیب می‌کنند. مظفری[۸] و همکاران

(2005) به کمک ویژگی‌های ساختاری و آماری، نقاط پایانی و نقاط محل تقاطعِ اجزای حروف در اسکلت کلمه را یافتند و سپس از آن‌ها در تقطیع کلمه به اجزای بنیادی آن استفاده کردند. ابراهیمی[۹] و کبیر (2008) از جداسازی حروف صرف‌نظر کرده و با یک روش کل‌گرا و گرافیکی صرفاً به بازشناسی زیرکلمات پرداخته‌اند. خسروی[۱۰] و کبیر (2009) برای جداسازی حروف از یک الگوریتم ترکیبی استفاده کرده‌اند که در آن ابتدا کلمهٔ خام را به عناصر تشکیل‌دهندهٔ آن تجزیه کردند و سپس با پردازش و ترکیب مکرر عناصر زیرحرفی از پس و پیش سعی در بازشناسی حروف داشتند. در این روش هر جا که ترکیبات عناصر نامفهوم‌اند، از روش کل‌گرا و مقایسهٔ آن با کلمات و زیرکلمات موجود در واژه‌نامه استفاده می‌شود. پژوهشگران بسیاری ازجمله شفیعی[۱۱] (2014)، اساساً به‌دلیل دشواری‌های ناشی از وجود خط واصل، از جداسازی حروف صرف‌نظر و صرفاً به بازشناسی زیرکلمات اکتفا می‌کنند. مسکنتی[۱۲] و کشاورز[۱۳] (2017) برای تشخیص درست کلمات به پس‌پردازش واژگانی متون بدخوانده‌شده به الگوریتم‌های زبان‌شناختی متوسل شدند. الگوریتم پیشنهادی کیایی[۱۴] و همکاران (2019) با آن‌که صرفاً بر روی متون چاپی تک‌فونتی محدودی اعمال شده است، نتایج مطلوبی نداشته و در برابر هم‌پوشانی حروف ناکارآمد است. تحقیق رحمتی[۱۵] و همکاران (2020) آخرین پژوهش بر مبنای جداسازی حروف بوده است. آن‌ها نیز مانند سایر پژوهش‌های موجود، خط واصل را جزو حرف اصلی محسوب کرده و الگوریتم پیشنهادی آن‌ها صرفاً برای کوتاه کردن کشیدگی‌های بیش از حد خط واصل در جهت سهولت بازشناسی حروف توسط نظام‌های موجود بوده است. سایر تحقیقات جدید از جداسازی حروف براساس ویژگی‌های ساختاری حروف اجتناب کرده و عمدتاً از الگوریتم‌های کل‌گرا و مبتنی بر شبکه‌های عصبی در جهت استخراج ویژگی‌های متمایز استفاده کرده‌اند (Bonyani & Jahangard, 2020).

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

درنتیجه، بازشناسی حروف فارسی و عربی به‌دلیل ویژگی‌های منحصربه‌فرد و تفاوت بنیادی آن‌ها با حروف لاتین با مشکلات بسیاری روبه‌رو بوده است و پس از گذشت چندین دهه از اولین چالش‌ها، با آن‌که برنامه‌های سایر خطوط به حد کمال رسیده و سال‌هاست وارد بازار شده‌اند، برنامه‌های فارسی و عربی هنوز کارآیی لازم را ندارند. در این میان، با وجود میلیون‌ها نسخهٔ خطی، برنامه‌های بازشناسی حروف فارسی دست‌نویس نیز هنوز در مراحل مقدماتی خود قرار دارند (Shafii, 2014) و متأسفانه با وجود شیوه‌های بسیار پیشرفتهٔ امروزی برای پردازش تصاویر متون لاتین‌محور، این شیوه‌ها عملاً به دلایل مربوط به «سبک» نگارش در پردازش تصاویر متون فارسی و عربی دچار مشکل می‌شوند (Pourreza et al., 2020).

۴. سیستم‌های اُ.سی.آر فارسی و مشکلات آن

متأسفانه ، از آنجا که در بازشناسی حروف، کانون توجه و الگوی اصلی برنامه‌نویسان اولیه، خط لاتین و نیز چینی و ژاپنی و مشخصه‌های آن بوده است، برنامه‌نویسان بازشناسی حروف فارسی/عربی نیز، بدون توجه به اختلافات ماهوی بین خط فارسی و لاتین، از همان الگوریتم‌های مورد استفاده در بازشناسی حروف لاتین تقلید کرده‌اند. درنتیجه، عدم توجه کامل به ویژگی‌های بنیادی خط فارسی و اختلاف آن با خط لاتین، باعث شده تا کار بازشناسی حروف فارسی دچار پیچیدگی‌های ساختاری شود و نتواند پابه‌پای خط لاتین پیشرفت کند.

برای بسط و روشن ساختن این پیچیدگی‌ها، آشنایی با مفاهیم کاربردی و همچنین فراهم آوردن مقدمات لازم برای معرفی الگوریتم پیشنهادی ما در مرحلهٔ ذخیره‌سازی و مقایسهٔ الگوها در فرایند بازشناسی نوری حروف فارسی، لازم است ابتدا به شرح ویژگی‌های سیستم نگارشی فارسی پرداخته شود.

۵. تاریخچه، ویژگی‌ها و عناصر شکلی خط فارسی

نخستین کوشش نظام‌مند برای ایجاد صورت‌های نمادین حرفی به‌منظور بازنمایی واج‌های زبان فارسی به دوران هخامنشیان باز می‌گردد. در آن دوران، کاتبان با استفاده از عناصر اصلی خط ایلامی که نوعی از خطوط میخی محسوب می‌شود، الفبای نیمه‌هجایی فارسی باستان را خلق

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

کردند. در همین دوران استفاده از خط آرامی نیز چنانکه از گوشهٔ پایینی کتیبهٔ داریوش اول در نقش رستم برمی‌آید، در میان کاتبان رایج بوده است. به همین دلیل است که در دوره‌های بعدی نیز عناصر خط آرامی که یک خط ناپیوسته بوده، منشأ ساخت خطوط ایرانی در دوره‌های میانه است.

خط فارسی/ عربی برخلاف خط آرامی که همانند خط لاتین و با حروف مجزا نوشته می‌شده، خطی سرهم و پیوسته است که برخلاف نظر عموم، همچون خط مانوی نتیجهٔ مستقیم و نزدیک‌ترین خویشاوند خط پهلوی کتابی است. عناصر اصلی خط فارسی/ عربی (همانند خط واصل، دندانه، حلقه، دنباله[۱۶] و...) با خط پهلوی کتابی یکسان بوده و همانند خط پهلوی کتابی دارای گونه‌های با نقطه و بی‌نقطه است. خط فارسی/عربی همچنین همانند خط پهلوی کتابی از راست به چپ نوشته می‌شود. برای مشاهدهٔ عناصر مشترک این دو خط به جدول ۱ رجوع شود.

جدول ۱: شباهت عناصر اصلی الفبای فارسی/ عربی با عناصر اصلی الفبای پهلوی کتابی

Table 1: The similarity of Persian/Arabic main alphabetical features with witten Pahlavi main alphabetical features

الفبای فارسی مشابه	ارزش آوایی پهلوی	الفبای پهلوی کتابی	الفبای فارسی مشابه	ارزش آوایی پهلوی	الفبای پهلوی کتابی
ـ(ک، گ، ک گ ی)	*z*	ک	ـ(ر)	*b*	ـل
سـ(ق)	*s*	ـفـ ، ـوـ	ـ(ن، ق، س، ش)	*p*	ـو
ش، (س)	*ʃ*	ـلـ	جـ(ف ق)	*y, g, d*	ـف ـن ـب ـت ـژ ـذ
مـ م	*m*	ـمـ ـمـ	یـ (بـ پـ تـ ثـ)	*y, g, d*	ـر ـز ـب ـت ـی ـژ
أ ، إ ، (ا ، آ)	*r, n, w, ʕ*	ا	چ، ج، (خ، ح)	*ʧ, ʤ, ʒ*	ـچ
ق، (و)	*k, γ*	و	ط، ظ، ص، ض	*t, ɵ, ð*	ـط
غ، (ع،ح،خ،چ،ج)	*γ*	ـع	ل، لـ	*l*	ـل

خط فارسی دارای ۳۲ حرف اصلی (ا [أ،إ،آ]، ب، پ، ت، ث، ج، چ، ح، خ، د، ذ، ر، ز، ژ، س، ش، ص، ض، ط، ظ، ع، غ، ف، ق، ک، گ، ل، م، ن، و [ؤ]، ه [ۀ، ة]، ی [ئ، ي]) بوده که از ۲۸ حرف الفبای عربی به‌علاوه چهار حرف دیگر (گ، چ، پ، ژ) تشکیل شده است. اما به‌علت تلاقی فرهنگی اعراب و ایرانیان در دوره‌های اسلامی و ورود آیات و احادیث و کلمات عربی فراوان، متون فارسی به

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

آمیزه‌ای از خطوط عربی و فارسی تبدیل شده است. اگرچه عناصر اصلی این دو خط یکی است، اما به‌دلیل عدم وجود حروف مشخص برای مصوت‌های کوتاه و مشکل تلفظ صحیح کلمات، عناصر فرعی همچون اعراب و نقطه‌گذاری زبان عربی نیز به فارسی راه یافته است. جدول ۲ حروف الفبای فارسی/عربی و حالات مختلف آن‌ها را در آغاز، میان، و پایان کلمه نشان می‌دهد.

جدول ۲: حروف الفبای فارسی/ عربی

Table 2:Persian/Arabic alphabets

٥	عنصر اصلی	حروف الفبای فارسی/عربی و حالات مختلف آن در کلمه			
		حالت تنها	حالت پایانی	حالت میانی	حالت ابتدایی
1	ب	ب	ـب	ـبـ	بـ
	پ	پ	ـپ	ـپـ	پـ
	ت	ت	ـت	ـتـ	تـ
	ث	ث	ـث	ـثـ	ثـ
2	ن	ن	ـن	ـنـ	نـ
3	ی	ی	ـی	ـیـ	یـ
	ي	ي	ـي		
	ئ	ئ	ـئ	ـئـ	ئـ
4	س	س	ـس	ـسـ	سـ
	ش	ش	ـش	ـشـ	شـ
5	ف	ف	ـف	ـفـ	فـ
6	ق	ق	ـق	ـقـ	قـ
7	ح	ح	ـح	ـحـ	حـ
	ج	ج	ـج	ـجـ	جـ
	چ	چ	ـچ	ـچـ	چـ
	خ	خ	ـخ	ـخـ	خـ
8	ص	ص	ـص	ـصـ	صـ
	ض	ض	ـض	ـضـ	ضـ
9	ط	ط	ـط	ـطـ	طـ
	ظ	ظ	ـظ	ـظـ	ظـ
10	ع	ع	ـع	ـعـ	عـ
	غ	غ	ـغ	ـغـ	غـ
11	م	م	ـم	ـمـ	مـ
12	ه	ه	ـه	ـهـ	هـ
	ة	ة	ـة	٥	٥
	ۀ	ۀ	ـۀ	٥	٥
13	ک	ک	ـک	ـکـ	کـ
	گ	گ	ـگ	ـگـ	گـ
14	ل	ل	ـل	ـلـ	لـ
15	ا	ا	ـا	٥	٥
	آ	آ	ـآ	٥	٥
	أ	أ	ـأ	٥	٥
	إ	إ	ـإ	٥	٥
16	د	د	ـد	٥	٥
	ذ	ذ	ـذ	٥	٥
17	ر	ر	ـر	٥	٥
	ز	ز	ـز	٥	٥
	ژ	ژ	ـژ	٥	٥
18	و	و	ـو	٥	٥
	ؤ	ؤ	ـؤ	٥	٥

عناصر شکلی خط فارسی را می‌توان به صورت زیر طبقه‌بندی کرد:

الف) عناصر دستهٔ اول عبارت‌اند از ۱۸ نویسهٔ اصلی (نویسهٔ تنها یا مجزا)، به همراه اشکال فرعی این نویسه‌ها (یعنی نویسه‌های آغازین، میانی، و پایانی) که به کمک خط واصل (نک. شکل ۱) و بر روی خط زمینه با هم ترکیب شده و باعث خلق ترکیبات دیگری می‌شوند: [ا (ا)، ب (ـب، ـبـ، بـ)، ن (ـن، ـنـ، نـ)، ی (ـی، ـیـ، یـ)، ح (حـ، ـحـ، ـح)، د (ـد)، ر (ـر)، س (سـ، ـسـ، ـس)، ص (صـ، ـصـ، ـص)، ط (طـ، ـطـ، ـط)، ع (عـ، ـعـ، ـع)، ف (فـ، ـفـ، ـف)، و (ـو)، ک (کـ، ـکـ، ـک)، ل (لـ، ـلـ، ـل)، م (مـ، ـمـ، ـم)، و (ـو)، ه (هـ، ـهـ، ـه)]. در نگارش فارسی به‌دلیل چسبیدگی نویسه‌ها، حروف از یک یا دو طرف به حروف مجاور خود اتصال دارند و برخی هم به صورت مجزا نوشته می‌شوند. بنابراین مرز حروف مشخص نیست و هر کلمه ممکن است شامل یک حرف مجزا یا چند حرف متصل باشد که «زیرکلمه»[۱۷] نامیده می‌شود. به‌عنوان مثال، واژهٔ «جـا مـه دا را ن»، یک کلمه، با ۹ حرف، اما ۷ زیرکلمه است. این چسبیدگی یا سرهم‌نویسی، بازشناسی متون فارسی را برای نظام‌های اُسی.آر بسیار مشکل می‌سازد.

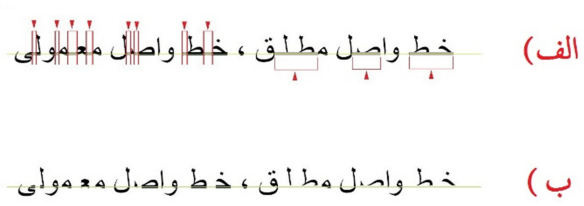
ب) عناصر دستهٔ دوم یعنی نقطه‌ها (یک نقطه، دو نقطه، سه نقطه و چهار نقطه)، همزه (ء) و مد (~) که عناصر اولیه به کمک آن‌ها به حروف تبدیل می‌شوند.

ج) عناصر دستهٔ سوم از اجزای اعراب‌گذاری هستند که در تغییر حرف نقشی ندارند و شامل فتحه، کسره، ضمه، تشدید، سکون، تنوین‌ها و علامت‌های تجویدند. عناصر اعراب‌گذاری در بالا یا پایین حروف قرار گرفته و در بیشتر موارد معنای کلمه را عوض می‌کنند. اما علامات تجوید که در متون قرآنی به‌کار می‌روند، تنها بر شیوهٔ خواندن و تلاوت جمله‌ها و کلمات دلالت می‌کنند و ظاهر آن‌ها بیشتر شباهت به حروف بالانویس[۱۸] در خط لاتین دارد.

د) برجسته‌ترین ویژگی و تفاوت خط فارسی با بیشتر خطوط دیگر استفادهٔ آن از عنصر «خط واصل» یا «واصل» است که بر روی خط زمینه قرار گرفته است و اغلب حروف را به هم وصل می‌کند. این عنصر نشئت‌گرفته از خط پهلوی کتابی بوده و عامل اصلی پیوستگی بین حروف است. در برنامه‌های طراحی نوع قلم، در صورتی که خط واصل بیش از حد متعارف امتداد یابد به نام «کشیده» یا «تطویل»[۱۹] خوانده می‌شود. دلیل این نام‌گذاری این است که محققان هیچ‌گاه خط واصل را به‌عنوان عنصری جداگانه به حساب نیاورده‌اند، بلکه آن را جزئی از حرف اصلی و

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

امتداد آن می‌دانسته‌اند. خط واصل نشان‌دهندهٔ پیوستگی حروف در زیرکلمات مرکب است. در این زیرکلمات، تمام حروف با کمک خط واصل به هم متصل می‌شوند. خط واصل را می‌توان به دو شیوهٔ مطلق و معمولی مورد بررسی و مطالعه قرار داد: (نک. شکل ۱).



شکل ۱
Figure 1

شکل ۱-الف محدودهٔ خط واصل را نشان می‌دهد. شکل ۱-ب ، شکل ظاهری کلمه‌ها را پس از حذف خط واصل نشان می‌دهد. «خط واصل مطلق» دربرگیرندهٔ آن قسمت از زیرکلمه است که به طور افقی و در راستای خط زمینه قرار دارد. اما، «خط واصل معمولی» محدود به آن قسمت از خط واصل مطلق است که هیچ پیکسل سیاهی بالا و پایین آن وجود ندارد. عنصر خط واصل به همراه عنصر فاصله باعث می‌شوند تا بیشتر حروف فارسی چهار حالت (اول، وسط، آخر، و مجزا) را دارا باشند.

ه) فاصله کوتاه (فاصلک)، عنصر دیگری است که بر روی خط زمینه جای دارد و زیرکلمات را از هم جدا می‌کند. این فاصله با فاصله‌ای که به نام space معروف است، تفاوت دارد و از آن بسیار کوتاه‌تر است. ازاین‌رو، عملکرد آن تنها نشان دادن عدم پیوستگی حروف و جدا کردن زیرکلمات از یکدیگر است. در متون جدید که با استفاده از پردازش‌گرهای الکترونیکی نوشته شده، اندازهٔ این فاصلهٔ کوتاه که بین زیرکلمات است با اندازهٔ فاصلهٔ بلند که بین کلمات وجود دارد تفاوت دارد. اما در متون قدیم‌تر این دو فاصله همیشه حفظ نشده‌اند و درنتیجه، تنها با اعمال فرایند پس‌پردازش بر این متون می‌توان تفاوت این دو را ثبت کرد. شناخت این ویژگی‌ها باعث می‌شود تا برنامه‌نویس صرفاً به مقایسهٔ تصاویر برگرفته از متن با الگوهای موجود نپردازد و

قبل از مبادرت به انجام مقایسه، دست به پردازش و دسته‌بندی صحیح اجزای آن بزند.

در اینجا، در راستای چنین هدفی به دنبال فراهم آوردن مقدمات و شرایط تولید الگوریتم ویژه‌ای هستیم که در بخش متن‌خوانِ خودکار و یا همان، بازشناسی نوری نویسه‌ها، ارائه می‌شود. روشن است که این مهم، با ویژگی و قابلیت‌های فراوان و ذاتی دستگاه خط فارسی، هم‌سوست.

همچنین چهره‌ای متفاوت از دستگاه خط فارسی به هنگام کار با رایانه‌ها و به‌ویژه در نظام‌های بازشناسی نوری حروف ارائه می‌دهیم که مخاطب عملاً به این نتیجه خواهد رسید که ظرفیت‌های دستگاه خط فارسی چیزی از دستگاه خط لاتین کم ندارد و اگر هم کاستی و کوتاهی در جایی احساس می‌شود زاییدهٔ عدم شناخت ماهیت خط فارسی است و ذاتی دستگاه خط نیست.

۶. مراحل بازشناسی نوری حروف فارسی و چالش‌های مربوطه

در این بخش، مخاطب با مراحل کلی بازشناسی نوری حروف و چالش‌های مربوطه در رویارویی با تصاویر متون فارسی آشنا می‌شود و در هر مرحله مشخص خواهد شد که ویژگی‌های خط فارسی چگونه به مقابله یا کمک برنامه‌نویس می‌آید. در این روند، تفاوت‌های روش پیشنهادی این پژوهش با روش‌های معمول در مراحل مختلف بازشناسی حروف نیز بیان شده است.

به‌طور کلی، هدف از انجام بازشناسی، تبدیل متن خام یعنی متنی که مطلقاً به‌صورت تصویری وجود دارد و قابل ویرایش نیست به متنی است که همانند متن تایپ‌شده قابل جست‌وجو و ویرایش باشد (Bonyani & Jahangard, 2020). در همین راستا، منظور از حرف خام، کلمهٔ خام و زیرکلمهٔ خام، نیز تصویر آن حرف، کلمه و زیرکلمه است، که گرچه برای مخاطب آشنا با آن خط قابل خواندن است، اما برای رایانه فقط به‌صورت یک تصویر، که مشتمل بر خانه‌ها یا پیکسل‌های نوری است، قابل درک است. به همین سبب است که اولین مرحله در بازشناسی نوری، پیش‌پردازش نام دارد. در این مرحله، ابتدا چرخش، کجی و نویز موجود در تصویر خام به کمک ابزارهای گرافیکی و الگوریتم‌های مختلف تصحیح می‌شود. سپس، تصویر متن به‌طور یکدست به‌صورت سیاه و سفید درمی‌آید تا نرم‌افزار بتواند آن را به‌صورت صفر (پیکسل سفید) و یک

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

(پیکسل سیاه) درک کند و توانایی پردازش سریع داده‌های آن را داشته باشد. شرح آخرین پیشرفت‌های مربوط به این حوزه در احمد[۲۰] (2018, pp.82-106) آمده است. خسرو بیگی[۲۱] و همکاران (2020) نیز روشی جدید برای حذف نویز و عناصر غیرمتنی از تصویر خام ارائه کرده‌اند که مبتنی بر پنج ویژگی خطی است و هم‌زمان با مرحلهٔ بازشناسی حروف و کلمات عمل می‌کند. پس از اصلاح گرافیکی متن خام، مراحل اصلی پردازش به شرح زیر است:

الف) **تشخیص عناصر صفحه و تقسیم‌بندی آن به چارچوب یا بلوک‌های متفاوت**: در این مرحله، خطوط زمینه بازشناسی شده و صفحه به قطعات یا بلوک‌های مختلف ازقبیل شکل، نمودار، جدول، و متن تقسیم‌بندی می‌شود (Alghamadi & William, 2019). یکی از روش‌های متداول برای این کار، مقایسهٔ میزان تراکم پیکسلی عمودی و افقی است. توضیح این‌که خطوط زمینه دارای بیشترین تراکم افقی و مرز بین بلوک‌ها دارای کم‌ترین تراکم افقی و عمودی هستند. در این مرحله، در صورتی که عکس وارونه یا سر و ته باشد، می‌توان با به‌کارگیری یکی دیگر از ویژگی‌های خط فارسی، که در آن تراکم پیکسلی عناصر بالای خط زمینه بیشتر از تراکم مشابه در پایین خط زمینه است، جهت درست متن را نیز تشخیص داد. البته، ویژگی‌های دیگری همچون از راست‌نویسی و در نتیجهٔ آن فضای خالی در سمت چپ و آخر پاراگراف‌ها نیز می‌تواند به‌عنوان معیار مورد استفاده قرار گیرد، اما این ویژگی همیشه در دسترس نیست.

ب) **جداسازی تصاویر خطوط درون هر بلوک متنی**: در این مرحله، پردازش هریک از این بلوک‌ها به‌طور جداگانه صورت می‌پذیرد و این بلوک‌ها به پاراگراف‌ها، خطوط، جملات، کلمات، زیرکلمات، و درنهایت به حروف جداگانه تقسیم‌بندی می‌شوند (Alghamadi & William, 2019). در اینجا نیز با استفاده از کم بودن تراکم پیکسلی افقی، مرز بین خطوط مشخص شده و تصاویر این خطوط از هم جدا می‌شوند. روش‌های مختلف در فرایند جداسازی اجزای صفحه به تفصیل در شفیعی (2014) آمده است. تا اینجا، تمام نظام‌های اُ.سی.آر کمابیش موفق عمل کرده و تفاوت‌های اصلی آن‌ها در مرحلهٔ جداسازی و بازشناسی کلمات است.

ج) **جداسازی کلمات و زیرکلمات**: در این مرحله، هریک از این خطوط خام، براساس عدم

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

تراکم پیکسلی عمودی و افقی، به زیرکلمه‌های خام تقسیم‌بندی شده و برای مرحلهٔ بعدی (که نوع فرایند آن بستگی به ذهنیت برنامه‌نویس دارد) آماده می‌شوند. در اینجا، براساس کوتاهی و بلندی فاصله‌های موجود بر خط زمینه، ابتدا، با توجه به فاصله‌های بلند، کلمات از یکدیگر جدا شده و سپس، با توجه به فاصله‌های کوتاه (یا فاصلک‌ها)، دو دسته از تصاویر خام از هم جدا می‌شوند: ۱) زیرکلمات تکی یا حروف تنها؛ ۲) زیرکلمه‌های مرکب (متشکل از حروف آغازین، میانی و پایانی). در برنامه‌های ویژهٔ لاتین، به‌دلیل وجود فاصلک بین حروف و فاصلهٔ بین کلمات (و همچنین فقدان عنصر زیرکلمه) کار جداسازی کلمات و سپس حروف از هم به آسانی انجام می‌پذیرد و در مرحلهٔ بعدی با مقایسهٔ حرف خام با الگوهای حرفی، همهٔ حروف به سهولت بازشناسی و به متن قابل ویرایش تبدیل می‌شوند. اما همهٔ برنامه‌های موجود، هنگام کار با خط فارسی، حتی با فرض این‌که حروف هم‌پوشانی نداشته و متن موجود دارای اعراب‌گذاری نبوده و همچنین این برنامه‌ها همگی توانایی بازشناسی حروف مجزا را داشته باشند، در تشخیص حروف زیرکلمه‌های مرکب با مشکل اساسی مواجهه می‌شوند.

د) جداسازی و **بازشناسی عناصر کلمه**: پژوهشگران شیوه‌های جداسازی و بازشناسی کلمات را به‌طور کلی به دو رویکرد کل‌گرا[۲۲] (غیرتفکیکی) (که در آن کلمات یا زیرکلمات خام به کمک یک واژه‌نامهٔ حجیم تصویری بازشناسی می‌شوند) و جزءگرا[۲۳] (تفکیکی) (که در آن حروف یا زیرحروف با کمک یک حرف‌نامهٔ تصویری کم‌حجم‌تر بازشناسی می‌شوند) تقسیم‌بندی می‌کنند (Alghamadi & William, 2019). در اینجا علاوه‌بر شرح این رویکردها، به بررسی عیوب آن‌ها نیز می‌پردازیم:

۱) در رویکرد کل‌گرا یا غیرتفکیکی، با تبعیت از شیوهٔ جداسازی در سیستم‌های لاتین‌محور، کل کلمه یا فقط زیرکلمه‌ها به‌عنوان الگویی واحد درنظر گرفته می‌شود. با توجه به وابستگی این روش به یک واژه‌نامهٔ تصویری با حجم بالا، شامل کلمات یا زیرکلمات، در قلم‌ها و دستخط‌های مختلف، رویکردهای کل‌گرا صرفاً زمانی دارای کارایی بسیار بالایی هستند که برای بازشناسی متون محدود به واژه‌های خاص (همانند: فرم‌های بانکی و مانند آن) به‌کار روند (Choudhary, 2014). این روش، به‌سبب ارائهٔ نتایج موفقیت‌آمیز در بازشناسی متون آشنا، در سال‌های اخیر موضوع پژوهش‌های بسیاری قرار گرفته است. اما در این نوع روش‌ها، همانند خسروبیگی و

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

همکاران (2020)، اگر متن خام دارای ترکیبات جدید ثبت‌نشده در واژه‌نامه (مثل واژه‌های جدید و سایر ترکیبات شامل حروف و عدد، ازقبیل، ۲-الف، ۵م، A2) باشد، این ترکیبات به‌عنوان نویز و عناصر غیرمتنی محسوب شده و از متن حذف می‌شوند. عیب دیگر این شیوه نسبت‌به شیوهٔ تفکیکی این است که هرچه تنوع زبان، قلم و دست خط‌ها در متن خام بیشتر شود، حجم واژه‌نامه (یا مجموعه داده‌های تصویری مشتمل بر الگوهای ازپیش تعریف‌شده) و حجم محاسبات مربوط به مقایسهٔ الگوها بیشتر می‌شود و در این صورت، علاوه‌بر اتلاف وقت، نتیجهٔ چندان رضایت‌بخشی نیز نخواهد داشت.

۲) در رویکرد جزءگرا یا تفکیکی، تلاش بر این است که کلمات خام درنهایت به حروف تجزیه و بازشناسی شوند. این رویکرد به دو نوع «تفکیک صریح»[۲۴] و «تفکیک ضمنی»[۲۵] تقسیم‌بندی می‌شود. در شیوهٔ تفکیک صریح، سیستم از ابتدای خط شروع کرده و متن خام را به عناصر کوچک‌تری به‌نام حروف («تفکیک مستقیم») یا زیرحروف (همانند دایره، پاره‌منحنی، خط راست و ...) («تفکیک غیرمستقیم») قطعه‌بندی می‌کند و سپس تلاش می‌شود تا با شناسایی زیرحروف و ترکیب آن‌ها ابتدا حروف و سپس کلمه بازشناسی شود (Alghamadi & William, 2019). در این روش، تشابه بسیاری از عناصر مشترک بین حروف فارسی و اشتباه در شناسایی مرز بین حروف یا زیرحروف از قبل، باعث می‌شود تا سیستم، هنگام عدم توانایی در تشخیص صحیح یک حرف براساس اجزای آن، در تشخیص حرف بعدی نیز دچار اشتباه و سردرگمی شود. برای مثال، اگر سیستم دو حرف اول از کلمه (ملیت) را، به‌دلیل تشابه عناصر نویسه (ـا = ـ + ا) با (ـلـ = ـ + ا + ـ) به‌صورت (ما) بازشناسی کند، آنگاه ناچار است، با صرف‌نظر کردن از عنصر خط واصل (ـ)، قسمت بعدی را نیز به‌صورت (یت) بخواند.

در روش تفکیک ضمنی، مراحل جداسازی و بازشناسی حروف همزمان انجام می‌شود و سیستم تصویر خام را برای عناصر ازپیش تعریف‌شده، جست‌وجو می‌کند (Alghamadi & William, 2019). اما عیب این رویکرد آن است که مرز بین حروف خام از قبل مشخص نشده و برنامه با توسل به الگوهای حرفی خود از ابتدای متن خام شروع به مقایسه می‌کند و گام به گام جلو می‌رود تا آن بخش خام جداشده به یکی از الگوهای حرفی موجود شباهت پیدا کند. آنگاه آن قسمت خام را به‌عنوان یک حرف محسوب می‌کند و به کناری می‌گذارد. سپس همین فرایند را

دوباره از ابتدای قسمت خام بعدی شروع می‌کند تا حرف جدیدی را بازشناسی کند. در این رویکرد، تقطیع زیرکلمات به حروف به روش ماشینی صورت می‌پذیرد که در بیشتر اوقات از دید انسانی غیرمنطقی است. مثلاً، برنامه‌هایی همانند ABBYY FineReader و Readiris، که از بهترین برنامه‌های موجود هستند، زیرکلمه‌ها را در گام‌ها یا فواصل مساوی ازقبل تعریف‌شده، برش می‌زنند. به‌دلیل عدم تساوی طول حروف در نگارش فارسی، این کار باعث می‌شود تا درصد بالایی از برش‌ها شامل بخشی از یک حرف، ترکیبی از یک حرف و بخشی از حرف مجاور، و یا ترکیبی از بخش آخر یک حرف و بخش اول حرف مجاور شوند که درنتیجه باعث بالا رفتن خطا در بازشناسی حروف می‌شود. متأسفانه مشکل به اینجا ختم نمی‌شود و در اغلب موارد قسمتی از حروف که به هر دلیل برای سیستم قابل تشخیص نیست، کاملاً حذف می‌شوند.

۷. استفاده از ویژگی‌های خط فارسی برای اصلاح سیستم‌های OCR

همانگونه که از نظر گذشت، تمام نظام‌های موجود در مرحلهٔ بازشناسی عناصر کلمه، دارای معایب اساسی است و بیش از آن‌که از ویژگی‌های خط فارسی و تفاوت‌های بنیادی آن با خطوط لاتین کمک بگیرند، به دیدهٔ مانع و کاستی به آن نگریسته‌اند. ازاین‌رو ما در اینجا با رویکردی جدید و منحصربه‌فرد به این ویژگی‌ها نزدیک شده و از آن‌ها در جهت ارتقا و اصلاح نظام‌های موجود بهره برده‌ایم.

الف) جهت نگارش فارسی، برخلاف بیشتر زبان‌ها، از راست به چپ است. درنتیجه، متأسفانه برنامه‌های اُ.سی.آر موجود، به‌علت تبعیت از الگوریتم‌های خط لاتین و همچنین امکان بازشناسی کلمات لاتین موجود در متون فارسی، هنوز متون فارسی را نیز از چپ به راست مورد تحلیل و پردازش قرار می‌دهند و این باعث می‌شود تا متون خام ابتدا به‌صورت برعکس و با خطوط درهم و گسسته بازشناسی شده و سپس با یک الگوریتم ساده و چند دستور تکمیلی، در بازده نهایی برنامه، یعنی متن تایپ‌شده، جملات به‌صورت از راست به چپ و پیوسته پدیدار می‌شوند. تا اینجای کار، به‌جز پیچیدگی الگوریتم‌های بازشناسی، مشکل چندانی ایجاد نمی‌شود، اما به‌دلیل چشم‌پوشی از جهت نگارش خط فارسی، کاربر فرضی در مرحلهٔ آموزش سیستم و به هنگام تعامل با آن نیز دچار مشکلات بنیادی می‌شود. برای نمونه، برنامه (سیستم) به جای نشان دادن

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

تصویر یک حرف به کاربر جهت شناسایی نویسه، در بیشتر اوقات تنها بخشی از یک حرف یا بخشی از دو حرف و ... را نشان می‌دهد که هیچ سنخیتی با حروف فارسی ندارند. درنتیجه، کاربر قدرت تعامل با نرم‌افزار را عملاً از دست می‌دهد و مرحلهٔ آموزش و تعامل با نرم‌افزار به مرحله‌ای تزیینی و ناکارآمد بدل می‌شود.

به‌منظور رفع این نقیصه، می‌توان ابتدا به کمک ویژگی‌های خط لاتین، کل متن خام را برای یافتن کلمات لاتین بازبینی کرد و پس از یافتن عناصر لاتین، فقط همان بخش از تصویر را برای ورود به الگوریتم بازشناسی لاتین کدگذاری کرد و جدا ساخت. سپس با به‌کارگیری هم‌زمان فرایند معکوس‌سازی تصویر متن و حذف عنصر خط واصل (جهت مشخص کردن مرز بین حروف خام)، الگوریتم‌های موجود را (که همگی ذاتاً برای خطوط از چپ به راست نوشته شده‌اند) با خط فارسی سازگار کرد. البته برای سهولت آموزش نرم‌افزار و تعامل با آن توسط کاربر فارسی‌زبان، می‌توان با افزودن چند دستور ساده، تصویر ارئه‌شده به کاربر را به‌صورت صوری به حالت اصلی بازگرداند تا کاربر دچار سردرگمی نشود. درنتیجه، این اصلاحات باعث می‌شوند تا بتوان به‌جای نوشتن الگوریتم‌های اساساً متفاوت و دستورات کاملاً جدید برای خط فارسی، همان امکانات نرم‌افزاری موجود برای بازشناسی خطوط لاتین را با اصلاحاتی کارآمد مجدداً مورد استفاده قرار داد.

ب) وجود چسبیدگی بین حروف: با آن‌که برخی از پژوهشگران، با توجه به اهمیت خط واصل، از آن به‌عنوان شاخصه‌ای برای جدا کردن حروف در زیرکلمه‌ها استفاده کرده‌اند، اما در روش پیشنهادی آن‌ها، علی‌رغم پیشرفت‌های موجود، علاوه‌بر مشکلات ناشی از عدم حذف خط واصل و هم‌پوشانی آن با عناصر دیگر (مانند نقاط و اعراب‌گذاری)، بلندی و کوتاهی این عنصر در قلم‌های مختلف نیز منشأ خطاهای متعددی شده است. از این گذشته، بسیاری از الگوهای فعلی دارای عنصر اصلی مشابه اما طول مختلف برای خط واصل هستند یعنی تفاوت شکلی آن‌ها، تنها در طول خط واصل است. حال آن‌که رایانه، به‌سبب تشخیص صرف خانه‌های سیاه و سفید، فقط می‌تواند با کمک نسبت و تناسب، تشابه تصاویر دارای طول و عرض متناسب را تشخیص دهد. برای نمونه، برنامهٔ آن‌ها، نه‌تنها «تـ» و «تـــ» را یک الگوی یکسان محسوب نمی‌کند، بلکه برای شکل «تـــ» اصلاً الگویی متناظر ندارد. برای تأیید این مطلب کافی است یک کلمهٔ ساده را به دو

شیوهٔ معمولی و کشیده بنویسید و تصاویر آن‌ها را به سیستم مذکور دهید، مشاهده خواهید کرد، که اولی را تشخیص می‌دهد، اما تصویر کشیدهٔ دوم را بازشناسی نخواهد کرد. ازاین‌رو، در رویارویی با این ویژگی، برنامه‌نویس باید با حذف خط واصل و گذاشتن یک کد ساده به‌جای آن (به‌منظور دسته‌بندی آتی نویسه‌ها) موجب شود تا هنگام مقایسهٔ الگوها، کشیدگی حروف و طول خط واصل تأثیری در بازشناسی حروف نداشته باشد. علاوه‌براین، در نتیجهٔ حذف خط واصل از الگوها، حجم هر الگو و تعداد کل الگوها نیز به‌طور قابل‌توجهی کاهش می‌یابد.

قطر خط واصل نیز دارای اهمیت ویژه‌ای است چون معمولاً ضخامت اصلی خط حروف (و همهٔ عناصر آن) با ضخامت خط واصل برابر است و می‌توان از این ویژگی برای تشخیص اندازهٔ قلم و یافتن تناسب صحیح بین حروف خام و الگوها استفاده کرد. در این صورت نه‌تنها لازم نیست، یک الگو را در اندازه‌های مختلف، ذخیره کرد، بلکه از حجم الگوها نیز کاسته می‌شود. مزیت دیگر خط واصل، تقسیم نویسه‌ها به چهار گروه (آغازین، میانی، پایانی، و تنها) است. این دسته‌بندی نیز خود به کمک فرایند بازشناسی می‌آید: به‌طوری که برخلاف بازشناسی حروف لاتین که معیار اصلی آن صرفاً مقایسهٔ حروف خام با الگوهای حرفی مجزاست، در بازشناسی حروف فارسی قادر خواهیم بود تا بر حسب خط واصل (و خط زمینه) که دربرگیرندهٔ عناصر اولیه است، تصاویر خام را به گروه‌های چهارگانهٔ فوق و همچنین به عناصر فوقانی و تحتانی طبقه‌بندی کنیم. البته، برای مقایسهٔ آن‌ها باید ابتدا الگوها را نیز طبقه‌بندی کرد و، برخلاف نظام‌های موجود، هریک از این دسته عناصر را فقط در مجموعه الگوهای همان گروه مورد مقایسه قرار داد وگرنه خطای بازشناسی و تشخیص حروف به شدت بالا می‌رود. مثلاً، تصویر خام حرف «ی» با «ک» بسیار شبیه بوده و در مرحلهٔ مقایسه با الگوها به راحتی با هم اشتباه می‌شوند. ازاین‌رو، تنها راه جلوگیری از این خطا آن است که به‌منظور جست‌وجو و یافتن برابرنهاده تصویر «ی»، فرایند مقایسه، برخلاف رویکرد فراگیر در خط لاتین، صرفاً با گروهی از الگوهای حرفی انجام شود که امتداد اعضای آن به زیر خط واصل (و نیز خط زمینه) می‌رود. درنتیجه، گروه‌بندی الگوها باید براساس معیار خط واصل انجام شود و همچنین تشخیص گروه عناصر خام باید براساس موقعیت آن‌ها نسبت‌به خط واصل و طبیعتاً پیش از فرایند مقایسهٔ عناصر خام با الگوها انجام شود تا برنامه بتواند در گروه خاصی به‌دنبال برابرنهادهٔ حرفی مورد نظر بگردد و از یک پردازش فراگیر و بیهوده جلوگیری شود. در این روش، دقت و سرعت عمل برنامه نیز بالا می‌رود و به

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

جای سازگار کردن خط فارسی با برنامه‌های رایج برای خط لاتین، فرایند بازشناسی با ویژگی‌های خط فارسی سازگار می‌شود.

به‌عنوان مثالی دیگر حرف «د» را درنظر بگیرید. این حرف فقط دارای دو شکل « ـد» (پایانی) و «د» (تنها) بوده و صرفاً در همین دو گروه قرار می‌گیرد. اما، از سویی، حرف خام «ح» نیز پس از حذف خط واصل به حرف «د» شبیه می‌شود و تنها راه جلوگیری از این خطا در بازشناسی آن است که این حرف خام به‌دلیل اتصال امتداد آن به خط واصل به‌طور خودکار در گروه حروف آغازین قرار بگیرد و درنتیجه، به‌دلیل عدم حضور «د» در گروه حروف آغازین هرگز امکان اشتباه آن با «ح» ایجاد نمی‌شود. ازسوی‌دیگر، حرف «ـد» نیز پس از حذف خط واصل به «د» شباهت دارد، اما به‌دلیل ثبت اولیهٔ این حرف خام، به‌دلیل وجود و موقعیت خط واصل، در گروه حروف پایانی، هرگز به‌صورت تنها بازشناسی نخواهد شد. از طرفی، چون عنصر خط واصل از تمام الگوهای مقایسه‌ای حذف می‌شوند، تنها راه بازشناسی صحیح بسیاری از حروف، به تشخیص حضور و موقعیت خط واصل در مراحل اولیهٔ برنامه (یعنی حذف آن از فرایند پردازش پس از کدگذاری و گروه‌بندی حرف خام) بستگی خواهد داشت. یکی دیگر از مزیت‌های فرایند حذف خط واصل به شیوهٔ بالا، عدم صرف وقت و انرژی برای مقایسهٔ این عنصر است که درصد بالایی از متون فارسی را تشکیل می‌دهد. بنابراین، رویکرد پیشنهادی ما، باعث می‌شود تا حداکثر استفاده از عنصر خط واصل صورت پذیرد و هم‌زمان، به‌دلیل حذف این عنصر پس از پردازش اولیهٔ آن و پرهیز از پردازش تکراری آن در مرحلهٔ مقایسه الگوها، از حجم پردازش کاسته شود.

ج) نقاط و اعراب‌گذاری در متون باعث می‌شود تا تراکم پیکسلی عمودی بیشتر شود و برنامه در تشخیص فاصلک (که عدم تراکم عمودی است) و خط واصل (که تراکم عمودی است) دچار مشکل هم‌پوشانی شود. روش پیشنهادی ما در اینجا، بر به‌کارگیری ابزار گرافیکی MWT[۲۶] استوار است. این ابزار باعث می‌شود تا برنامه با معیار قرار دادن خط زمینه و نقاط مشترک موجود بین عناصر اصلی کلمه و خط زمینه، قبل از تلاش برای تشخیص فاصلک‌ها، تمام عناصر غیراصلی (مانند نقاط و اعراب‌گذاری) را موقتاً حذف کند و بدین ترتیب احتمال خطای نرم‌افزاری را کاهش دهد. البته، پس از انجام موفقیت‌آمیز جداسازی عناصر اولیه، می‌توان نقاط و اعراب‌گذاری‌ها را که به‌طور جداگانه بازشناسی شده‌اند به عناصر اولیه اضافه کرد و با محاسبهٔ

سریع به ماهیت اصلی حروف پی برد. معیار قرار دادن خط زمینه در تعیین فاصلک‌ها همچنین باعث می‌شود تا بتوان حروفی را که (مانند: برج) در زیر خط به هم چسبیده یا هم‌پوشانی کرده‌اند از هم جدا کرد. در این صورت نیز می‌توان (به‌منظور عدم اشتباه نوک باقیمانده از حرف قبلی با نقطه) با قرار دادن کد ویژه‌ای بقایای حرف هم‌پوشان مجاور را از فرایند بازشناسی حذف کرد.

دو ویژگی مهم نقاط در حروف فارسی عبارت است از یکی این‌که، نقاط (با توجه به معیار خط واصل و راستای آن) به دو صورت فوقانی و تحتانی به‌کار می‌روند؛ دوم این‌که این نقاط تقریباً هیچ‌وقت کاملاً به عناصر اولیه نمی‌چسبند و همین باعث می‌شود بتوان به این روش از این ویژگی استفاده کرد: با انتساب دو کد ویژه به نقاط (یکی برای فوقانی و تحتانی بودن؛ دیگری برای طول و عرض آن به‌منظور تشخیص تعداد نقاط) و ضمیمهٔ این کدها به خروجی برنامه، پس از پردازش و تعیین عنصر اولیه از طریق مقایسهٔ الگوهای عناصر اولیه، می‌توان به سهولت با یک دستورسازی ساده، ماهیت حرف خام را تعیین و آن را به کاراکترهای قابل تشخیص برای یک واژه‌پرداز تبدیل کرد. در مواقعی که نقطه‌ای به حرفی چسبیده باشد نیز می‌توان با معیار قرار دادن ضخامت خط واصل و بهره‌گیری از روش‌های گرافیکی، آن را جدا کرد. عناصر مربوط به اعراب‌گذاری نیز همچون نقطه‌ها هیچ اتصالی به خط زمینه و عناصر اولیه ندارند، اما برخلاف نقاط که در راستای خط زمینه امتداد و تراکم دارند، این عناصر دارای شکل مورب هستند و در یک عرض ثابت تراکم ندارند و ارزش مقداری ستون‌های پیکسلی آن‌ها با پیش‌روی طولی پردازش کم‌تر یا زیادتر می‌شود. درنتیجه، در مرحلهٔ مقایسهٔ الگوها، به کمک شیب خط می‌توان به سرعت نقاط را از اعراب‌گذاری جدا کرد.

به‌طور خلاصه، با حذف خط واصل از تصویر متن و جایگزینی آن با یک کد ساده برخلاف آنچه مرسوم است یعنی جداسازی حروف براساس مقایسهٔ الگوها که باعث اشتباهات بی‌شماری می‌شود؛ ۱) حروف به‌طور طبیعی از هم جدا شده و کار جداسازی حروف با موفقیت انجام می‌شود؛ ۲) به‌جای مقایسهٔ یک حرف با همهٔ الگوهای موجود، به‌دلیل وجود کد خط واصل، فرایند مقایسه تنها بین شکل موجود (حرف خام) با الگوهای همان مجموعه صورت می‌پذیرد. به عبارتی، اشکال حروف به چهار دسته تقسیم خواهند شد: الف) شکل+ خط واصل (= حروف آغازین)؛ ب) خط واصل + شکل (= حروف پایانی)؛ ج) خط واصل+شکل+خط واصل (= حروف میانی)؛ د) شکل عادی که هیچ طرف آن خط واصل وجود ندارد (= حروف تنها). درنتیجه، به‌جای مقایسهٔ حرف

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

خام و تلاش برای انطباق آن با همهٔ الگوهای موجود، این مقایسه تنها بین حرف خام (شکل موجود) و الگوهای موجود در یکی از این چهار گروه اصلی صورت می‌پذیرد؛ ۳) با حذف خط واصل، کشیدگی حروف که در عمل تأثیری در خوانش کلمه ندارند، از فرایند مقایسه حذف می شود و سرعت پردازش افزایش می‌یابد؛ از طرفی، با معیار قرار دادن خط واصل و امتداد آن یعنی خط زمینه: الف) عناصر خام به دو دسته فرعی فوقانی و تحتانی تقسیم و در فرایند مقایسه نیز عناصر فوقانی فقط با الگوهای فوقانی و عناصر تحتانی فقط با الگوهای تحتانی قیاس می‌شوند؛ ب) هنگام مقایسهٔ عناصر اصلی با الگوها، به‌جای مقایسهٔ کلی کادر تصویر خام با الگوها، ابتدا خط زمینهٔ عنصر خام با خط زمینهٔ الگو تطبیق و سپس مقایسه در دو بخش فوقانی و تحتانی انجام می‌شود. درنتیجه، عناصر شکلی مشابه در بخش فوقانی هرگز با الگوهای بخش تحتانی تطبیق نخواهند شد. برای نمونه، دنبالهٔ «م» در پایین با «ـا» در بالا ، «ک» در بالا با «ی» در پایین، « ـَ » در بالا با « ـِ » در پایین تطبیق نمی‌شوند.

در بخش الگوبرداری و مقایسهٔ آن‌ها با عناصر خام نیز می‌توان از فرایند ویژهٔ انتقال هندسی فضای دوبُعدی به فضای یک‌بُعدی حداکثر استفاده را کرد.

۸. انتقال فضای دوبُعدی به تک‌بعدی

در حال حاضر، مقایسهٔ عناصر خام با الگوها، در فضای دوبُعدی (طول و عرض) و به صورت صفر و یک صورت می‌پذیرد. بدین معنی که هر پیکسل یا سیاه است (یعنی جزئی از نوشتهٔ خام که دارای کد ۱ است) و یا سفید (یعنی فضای نانوشته که دارای کد ۰ است). این مقایسه پیکسل به پیکسل یا خانه به خانه بوده و از گوشهٔ تصویر شروع می‌شود و نزدیک‌ترین الگوی تصویری با مختصات کمابیش مشابه را به آن منسوب می‌کند. سپس، برابرنهادهٔ حرفی مربوط به الگوی موردنظر به‌عنوان معادل حرفی آن عنصر درنظر گرفته می‌شود.

درنتیجه، تمامی نظام‌های اُ.سی.آر موجود در یک فضای دوبُعدی یعنی در یک صفحه کار می‌کنند و از آنجا که این‌گونه مدل‌های دوبعدی حجم محاسبات سنگینی دارند و مقایسه کردن الگوها نیز در آن‌ها مشکل است، مدل‌های مورد استفاده در پایگاه داده‌های آن‌ها پیچیده است و حجم پایگاه داده و الگوهای ذخیره‌شده در آن‌ها سنگین است. این مسئله به‌ویژه هنگامی که تعداد

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

قلم‌ها (فونت‌ها) افزایش می‌یابد به خوبی مشهود می‌شود، چراکه حجم اطلاعات قابل پردازش بالا می‌رود و بالطبع حجم محاسبات نیز افزایش می‌یابد و سرعت عملکرد الگوریتم به‌نسبت کاهش می‌یابد. بنابراین تقریباً تمامی نظام‌های اُ.سی.آر موجود کنونی به‌واسطهٔ عملکردشان که براساس مقایسهٔ کاراکترها با الگوهای دوبُعدی ذخیره‌شده در پایگاه داده‌هایشان است دارای ایراداتی هستند که به‌طور خلاصه می‌توان به دو مورد از مشکلات آنان اشاره کرد: الف) تأثیرپذیری بالا از پیوستگی و کشیدگی حروف فارسی و مشاهدهٔ درصد بالایی از خطا درمورد حروف به‌هم پیوسته و کشیده در این نرم‌افزارها؛ ب) حجم بالای اطلاعات و محاسبات به‌واسطهٔ الگوهای ذخیره‌شده و به دنبال آن کاهش سرعت و دقت این نرم‌افزارها.

پیشنهاد ما برای افزایش دقت عمل برنامه و کاستن از مدت زمان مقایسه و حجم داده‌ها، به‌کارگیری فرایند هندسی انتقال فضای دوبعدی به تک‌بعدی در پردازش متن خام و الگوبرداری است. اساس این فرایند تفاوت گرافیکی بین عناصر تشکیل‌دهندهٔ حروف است. از آنجا که تفاوت بین حروف و متمایز کردن آن‌ها در ذهن انسان تنها با توجه به شکل حرف صورت می‌گیرد و شکل حروف در فضای دوبُعدی صفحه و براساس طول و عرض امکان وجود می‌یابد، این فرایند با اتکا به طول و عرض پیکسل‌های قابل رؤیت یک حرف، که منحصر به همان حرف است، عمل می‌کند و هر حرف خام را به‌صورت یک گزارهٔ عددی در یک فضای تک‌بعدی نشان می‌دهد که معیار محاسبهٔ آن خط واصل است. در این روش، آن‌گونه که در ادامه خواهد آمد، با انتساب توان صفر به خط واصل، میزان کشیدگی آن، تأثیری بر شکل حرف و الگوی آن نخواهد داشت.

در این فرایند، ابتدا مطابق آنچه معمول است، تصاویر انتخابی را به یک ماتریس دوبُعدی تبدیل می‌کنیم به گونه‌ای که در این ماتریس هر پیکسل سفید را متناظر با یک درایه با مقدار صفر و هر پیکسل تیره را متناظر با یک درایه با مقدار یک درنظر می‌گیریم. برای مثال، یک تصویر فرضی با رزولوشن ۲۰×۱۰ پیکسل، به یک ماتریس ۲۰×۱۰ با مقادیر ۰ و ۱ تبدیل می‌شود. اما نقطهٔ افتراق و انحصاری الگوریتم پیشنهادی ما تبدیل این ماتریس دوبُعدی به یک ماتریس یک‌بُعدی و یا خطی منحصربه‌فرد با استفاده از روش ریاضی تبدیل باینری[۲۷] به دسیمال[۲۸] (تبدیل اعداد مبنای ۲ به ۱۰) است. در این روش، هر عدد طبیعی را می‌توان به‌طور منحصربه‌فرد به‌صورت مجموع توان‌های صحیح نامنفی عدد ۲ نمایش داد. برای نمونه، با استفاده از این روش

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

اعداد طبیعی زیر را می‌توان به‌صورت مجموع توان‌های صحیح نامنفی عدد ۲ نمایش داد:

$۲^{۲}+۲^{۱}+۲^{۰}=۷$ ، $۲^{۳}+۲^{۱}=۱۰$ ، $۲^{۵}=۳۲$ ، $۲^{۳}+۲^{۲}+۲^{۱}+۲^{۰}=۱۵$ ، $۲^{۶}+۲^{۵}+۲^{۲}=۱۰۰$

در اینجا، با به‌کارگیری روش مذکور، هر سطر ماتریس دوبُعدی را، با استفاده از اعداد صحیح نامنفی، ارزش‌گذاری می‌کنیم و درنتیجه، ارزش هر آرایه از ماتریس دوبعدی، با توجه به ارزش درنظر گرفته‌شده برای سطر n ام، از ضرب مقدار صفر یا یکِ آن آرایه در ۲ به توان ارزش‌های متناظر با آن سطر، یعنی (۲^{n}) به‌دست خواهد آمد. سپس، آرایه‌های هر ستون از ماتریس را با یکدیگر جمع کرده و در آرایه متناظر با آن قرار می‌دهیم. درنتیجه، ماتریس دوبعدی به یک ماتریس یک‌بعدی تبدیل می‌شود. این ماتریس به‌دست آمده با توجه به قضیهٔ ذکرشده، درحقیقت یک رابطهٔ دوطرفه منحصربه‌فرد است؛ بدان معنی که می‌توان در صورت نیاز، به سرعت، آرایهٔ یک‌بعدی را با عمل معکوس به ماتریس دوبعدی با مقادیر صفر و یک تبدیل کرد. مثال‌های زیر روشنگر مطالب مذکور است.

ماتریس ۷×٤ زیر را درنظر می‌گیریم:

ارزش توانی								ارزش مقداری
۰	۱	۰	۰	۰	۱	۰	۱	$۲^{۰}=۱$
۱	۱	۱	۰	۰	۱	۰	۰	$۲^{۱}=۲$
۲	۰	۱	۱	۱	۱	۰	۰	$۲^{۲}=۴$
۳	۱	۰	۱	۱	۱	۰	۱	$۲^{۳}=۸$

ارزش توانی سطر اول را صفر، سطر دوم را یک، سطر سوم را دو و سطر چهارم را سه درنظر گرفته و با انتخاب ارزش مقداری هر سطر به‌صورت ۲ به توان ارزش توانی آن سطر و ضرب آن در آرایه‌های صفر و یک ماتریس اولیه، ماتریس ارزش‌گذاری متناظر ماتریس بالا را به شکل زیر به‌دست می‌آوریم:

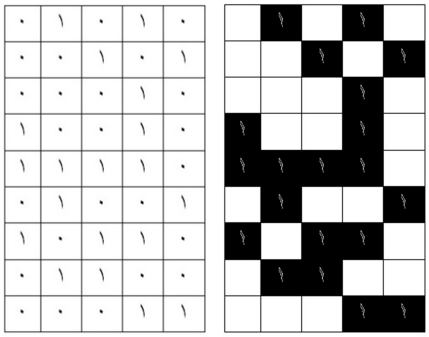
۱	۰	۱	۰	۰	۰	۱
۰	۰	۲	۲	۰	۲	۲
۰	۰	۴	۴	۴	۴	۰
۸	۰	۸	۸	۸	۰	۸

و درنهایت با جمع کردن مقادیر هر ستون این ماتریس، آن را به یک ماتریس یک‌بعدی تبدیل می‌کنیم:

۹	۰	۱۵	۱۴	۱۲	۶	۱۱

با این تبدیل ماتریس دوبعدی ۷×۴ به صورت منحصربه‌فرد به یک ماتریس یک‌بعدی ۷×۴ تبدیل می‌شود.

حال مثال بالا را به شکل یک ماتریس دوبعدی ۵×۹ تعمیم می‌دهیم



با قرار دادن ارزش توانی هر سطر به ترتیب از صفر تا هشت و انتخاب ارزش مقداری هر سطر به‌صورت ۲ به توان ارزش توانی آن سطر و ضرب آن در آرایه‌های صفر و یک ماتریس اولیه، ماتریس ارزش‌گذاری متناظر ماتریس بالا به‌شکل زیر به‌دست می‌آید

ارزش توانی						ارزش مقداری
۰	۰	۱	۰	۱	۰	$۲^{۰}=۱$
۱	۲	۰	۲	۰	۰	$۲^{۱}=۲$
۲	۰	۴	۰	۰	۰	$۲^{۲}=۴$
۳	۰	۸	۰	۰	۸	$۲^{۳}=۸$
۴	۰	۱۶	۱۶	۱۶	۱۶	$۲^{۴}=۱۶$
۵	۳۲	۰	۰	۳۲	۰	$۲^{۵}=۳۲$
۶	۰	۶۴	۶۴	۰	۶۴	$۲^{۶}=۶۴$
۷	۰	۰	۱۲۸	۱۲۸	۰	$۲^{۷}=۱۲۸$
۸	۲۵۶	۲۵۶	۰	۰	۰	$۲^{۸}=۲۵۶$

که درنهایت ماتریس یک‌بعدی متناظر با شکل بالا به‌صورت زیر حاصل می‌شود:

۲۹۰	۳۴۹	۲۱۰	۱۷۷	۸۸

در ادامه، شیوهٔ محاسباتی مذکور در قالب یک نمونهٔ منتخب از میان حروف الفبای فارسی و سپس یک کلمه نشان داده می‌شود. این ماتریس‌ها، در راستای ویژگی‌های خطی معرفی شده و دارای خصوصیات منحصربه‌فردی هستند:

۱) اندازهٔ ماتریس‌ها کاملاً اختیاری است و هیچ‌گونه تأثیری بر الگوریتم محاسبات ندارند.

۲) جهت استفاده از ویژگی‌های خط واصل و امتداد آن، ارزش خط زمینه در ماتریس‌های مذکور باید (۱) درنظر گرفته شود.

۳) ارزش سایر ردیف‌ها، به موقعیت آن‌ها نسبت‌به خط زمینه بستگی دارد. درنتیجه، خانه‌های فوقانی با ارزش توانی فرد و خانه‌های تحتانی با ارزش توانی زوج درنظر گرفته شده‌اند.

٤) و درنهایت این‌که در فضای واقعی کار با نرم‌افزار، سطح رزولوشن و وضوح تصاویر بسیار بالاتر بوده و در این مثال‌ها، هدف تنها نشان دادن بازدهی و کارآیی الگوریتم ارائه‌شده و نحوهٔ عملکرد آن است.

مثال ۱) حرف [ج]

شکل زیر پراکندگی خانه‌های سیاه و سفید را در ماتریس اولیه متعلق به این حرف نشان می‌دهد.

ردیف	ارزش توانی	ارزش مقداری	شماره ستون																	
			۱	۲	۳	٤	٥	٦	۷	۸	۹	۱۰	۱۱	۱۲	۱۳	۱٤	۱٥	۱٦	۱۷	۱۸	۱۹	۲۰
۱	۹	$۲^{۹}$																				
۲	۷	$۲^{۷}$																				
۳	٥	$۲^{٥}$				■	■	■	■	■	■	■										
٤	۳	$۲^{۳}$							■	■	■											
٥	۱	$۲^{۱}$								■	■	■										
٦	۰	$۲^{۰}$											■									
۷	۲	$۲^{۲}$							■				■									
۸	٤	$۲^{٤}$											■									
۹	٦	$۲^{٦}$										■	■									
۱۰	۸	$۲^{۸}$				■	■	■	■	■	■	■										
۱۱	۱۰	$۲^{۱۰}$																				

با درنظر گرفتن ارزش خانه‌های سیاه و سفید به ترتیب برابر با (۱) و (۰) ماتریس زیر حاصل می‌شود:

ردیف	ارزش توانی	ارزش مقداری	شماره ستون																	
			۱	۲	۳	٤	٥	٦	۷	۸	۹	۱۰	۱۱	۱۲	۱۳	۱٤	۱٥	۱٦	۱۷	۱۸	۱۹	۲۰
۱	۹	$۲^{۹}$																				
۲	۷	$۲^{۷}$																				
۳	٥	$۲^{٥}$			۱	۱	۱	۱	۱	۱	۱	۱										
٤	۳	$۲^{۳}$						۱	۱	۱												
٥	۱	$۲^{۱}$								۱	۱	۱										
٦	۰	$۲^{۰}$										۱	۱									
۷	۲	$۲^{۲}$							۱				۱									
۸	٤	$۲^{٤}$											۱									
۹	٦	$۲^{٦}$										۱	۱									
۱۰	۸	$۲^{۸}$			۱	۱	۱	۱	۱	۱	۱	۱										
۱۱	۱۰	$۲^{۱۰}$																				

سپس با اعمال شیوهٔ محاسباتی یادشده ماتریس ارزش‌گذاری زیر به‌دست می‌آید:

ردیف	ارزش توانی	ارزش مقداری	شماره ستون ۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰	۱۱	۱۲	۱۳	۱۴	۱۵	۱۶	۱۷	۱۸	۱۹	۲۰
۱	۹	$۲^{۹}$																				
۲	۷	$۲^{۷}$																				
۳	۵	$۲^{۵}$				۳۲	۳۲	۳۲	۳۲	۳۲	۳۲	۳۲										
۴	۳	$۲^{۳}$						۸	۸	۸												
۵	۱	$۲^{۱}$								۲	۲	۲										
۶	۰	$۲^{۰}$										۱	۱									
۷	۲	$۲^{۲}$							۴				۴									
۸	۴	$۲^{۴}$											۱۶									
۹	۶	$۲^{۶}$										۶۴	۶۴									
۱۰	۸	$۲^{۸}$				۲۵۶	۲۵۶	۲۵۶	۲۵۶	۲۵۶	۲۵۶	۲۵۶										
۱۱	۱۰	$۲^{۱۰}$																				

با جمع مقادیر هر ستون این ماتریس، ماتریس یک‌بعدی متناظر با حرف [ج] به شکل زیر به‌دست می‌آید:

۹۲	۳۳۲	۲۶۷	۲۶۷	۲۷۰	۲۶۶	۲۶۴	۲۶۴

مثال ۲) کلمهٔ «تبسم»

ردیف	ارزش توانی	ارزش مقداری	شماره ستون ۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰	۱۱	۱۲	۱۳	۱۴	۱۵	۱۶	۱۷	۱۸	۱۹	۲۰	۲۱	۲۲	۲۳	۲۴	۲۵	۲۶
۱	۹	$۲^{۹}$																										
۲	۷	$۲^{۷}$																										
۳	۵	$۲^{۵}$					■																					
۴	۳	$۲^{۳}$																					■	■	■			
۵	۱	$۲^{۱}$							■						■	■		■					■					
۶	۰	$۲^{۰}$																										
۷	۲	$۲^{۲}$																										
۸	۴	$۲^{۴}$								■																		
۹	۶	$۲^{۶}$																										
۱۰	۸	$۲^{۸}$																										
۱۱	۱۰	$۲^{۱۰}$																										

ردیف	ارزش توانی	ارزش مقداری	شماره ستون																								
			۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰	۱۱	۱۲	۱۳	۱۴	۱۵	۱۶	۱۷	۱۸	۱۹	۲۰	۲۱	۲۲	۲۳	۲۴	۲۵	۲۶
۱	۹	$۲^۹$																									
۲	۷	$۲^۷$																									
۳	۵	$۲^۵$			۱		۱																				
۴	۳	$۲^۳$																			۱	۱	۱	۱			
۵	۱	$۲^۱$			۱				۱					۱		۱		۱				۱					
۶	۰	$۲^۰$			۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	
۷	۲	$۲^۲$																							۱		
۸	۴	$۲^۴$								۱															۱		
۹	۶	$۲^۶$																							۱		
۱۰	۸	$۲^۸$																							۱		
۱۱	۱۰	$۲^{۱۰}$																									

باز هم با اعمال شیوهٔ محاسباتی قبل ماتریس ارزش‌گذاری زیر به‌دست می‌آید:

ردیف	ارزش توانی	ارزش مقداری	شماره ستون																								
			۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰	۱۱	۱۲	۱۳	۱۴	۱۵	۱۶	۱۷	۱۸	۱۹	۲۰	۲۱	۲۲	۲۳	۲۴	۲۵	۲۶
۱	۹	$۲^۹$																									
۲	۷	$۲^۷$																									
۳	۵	$۲^۵$			۳۲		۳۲																				
۴	۳	$۲^۳$																					۸	۸	۸		
۵	۱	$۲^۱$			۲				۲					۲		۲		۲				۲		۲			
۶	۰	$۲^۰$			۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	۱	
۷	۲	$۲^۲$																								۴	
۸	۴	$۲^۴$								۱۶																۱۶	
۹	۶	$۲^۶$																								۶۴	
۱۰	۸	$۲^۸$																								۲۵۶	
۱۱	۱۰	$۲^{۱۰}$																									

با جمع مقادیر هر ستون این ماتریس، ماتریس یک‌بعدی متناظر کلمهٔ «تبسم» به شکل زیر به‌دست می‌آید:

۳۵	۱	۳۳	۱	۳	۱	۱۷	۱	۱	۱	۳	۱	۳	۱	۳	۱	۱	۱	۱۱	۹	۱۱	۱	۱	۱	۳	۱	۳	۱	۱	۳۴۱

با به‌کارگیری مفهوم خط واصل در ساختمان الگوریتم پیشنهادی فوق، نویسه‌های متصله همگی دارای یک واحد خط واصل خواهند بود. بدین معنا که برای خط واصل یک واحد مبنا درنظر گرفته شده و این واحد مبنا کوچک‌ترین واحد ممکن برای خط واصل است و در هنگامی که خط واصل دارای بیش از این یک واحد مبنا باشد، کشیدگی‌های اضافی هریک از حروف نه با استفاده از روش‌های متداول پردازش تصویر بلکه با بهره‌گیری از الگوریتم‌ها و فرایندهای ریاضی و کدگذاری ویژه حاصل از آن حذف می‌شوند. بنابراین نرم‌افزار در مرحلهٔ مقایسه، کشیدگی‌های اضافی را حذف می‌کند و سپس با الگوهای ذخیره‌شدهٔ خود مقایسه را انجام می‌دهد.

۹. نتیجه

با نگاهی به وضعیت عملکردی نرم‌افزارهای اُ.سی.آر فارسی کنونی و با جمع‌بندی و دقت‌نظر در مسائل و مشکلات آن‌ها، به‌نظر می‌رسد دو چالش اصلی یکی در بخش مسائل مرتبط با خط فارسی و دیگری در بخش طراحی الگوریتم‌ها، و پایگاه‌های داده به‌کار رفته در این نرم‌افزارها فراروی محققان و متخصصان امر بوده است. در اینجا مشخص شد که سرمنشأ چالش‌های موجود، چشم‌پوشی برنامه‌نویسان از کارکرد اصلی و فلسفهٔ وجودی خط واصل و احتساب آن به‌عنوان جزئی از حرف اصلی و امتداد آن بوده است. سازندگان خط فارسی، مانوی و پهلوی کتابی به‌منظور ایجاد یک خط معیار برای راست‌نویسی در متن و حاشیهٔ صفحات بدون نیاز به کشیدن خط زمینه، و همچنین به‌منظور اتصال نویسه‌ها به یکدیگر، و نه احتساب آن به‌عنوان قسمتی از حروف متصله، دست به خلق این عنصر زدند. در اینجا، به‌منظور مرتفع ساختن چالش نرم‌افزارهای اُ.سی.آر فارسی در مرحلهٔ جداسازی زیرکلمات و در رویارویی با چسبندگی و یا سرهم بودن حروف و به‌طور ساده حساسیت آن‌ها به کشیدگی حروف، از ویژگی‌های برجستهٔ عنصر خط واصل و حذف صوری آن بهره‌برداری شد. خط واصل بخش عمده‌ای از متون

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

فارسی را تشکیل می‌دهد و در اینجا حذف صوری آن از متن خام و الگوها سبب شد تا از حجم پردازش و میزان خطاهای ایجادشده به‌واسطهٔ آن به شدت کاسته شود. همچنین، با معیار قرار دادن آن، امکان طبقه‌بندی عناصر خط فارسی و همچنین الگوها فراهم شد. درنتیجه، به‌جای مقایسهٔ یک عنصر خام با همهٔ الگوها، فرایند مقایسه به گروه‌های همگون محدود می‌شود و سرعت پردازش افزایش می‌یابد. درنهایت، الگوریتم پیشنهادی نیز بر پایهٔ خط واصل و امتداد آن بر خط زمینه و با بهره‌گیری از روش تبدیل داده‌های باینری به دسیمال میسر می‌شود. این الگوریتم با استفاده از فرایند انتقال فضای دوبعدی رایج به فضای تک‌بعدی، با تبدیل منحصربه‌فرد الگوها و عناصر خام از فضای دوبعدی صفحه به فضای تک‌بعدی خط گذر کرده و درنتیجه موجب می‌شود تا از حجم پایگاه داده‌ها به شدت کم و بر دقت و سرعت نرم‌افزار به‌طور قابل‌ملاحظه‌ای افزوده شود.

۱۰. پی‌نوشت‌ها

1. Optical Character Recognition (OCR)
2. Parhami
3. Taraghi
4. Azmi
5. Kabir
6. Menhaj
7. Adab
8. Mozaffari
9. Ebrahimi
10. Khosravi
11. Shafii
12. Maskanati
13. Keshavarz
14. Kiaei
15. Rahmati

۱۶. همانند دنبالهٔ حروف «ن، س، ق، ل».

17. subword
18. superscript
19. tatweel, kashida
20. Ahmad
21. Khosrobeigi

[DOR: 20.1001.1.23223081.1401.0.0.114.6]
[DOI: 10.29252/LRR.14.2.11]

22. holistic
23. analytical
24. explicit
25. implicit
26. Magic Wand Tool
27. binary
28. decimal

۱۱. منابع

- عاصی، س. م. (۱۳۸۵). فارسی در رایاسپهر جایگاه زبان فارسی در جهان نوین فناوری اطلاعات. *نامهٔ فرهنگستان*، ۸ (۳)، ۵۹ـ۷۰.

References

- Alghamdi, M., & William, T. (2018). Printed arabic script recognition: A Survey. *International Journal of Advanced Computer Science and Applications,* 9(9), 415-428.
- Aranian, M. J., Sarvaghad-Moghaddam, M., & Houshmand, M. (2017). Feature dimensionality reduction for recognition of Persian handwritten letters using a combination of quantum genetic algorithm and neural network. *Majlesi Journal of Electrical Engineering*, *11*(2), 1-6.
- Assi, S.M. (2006). Persian in cyberspace, position of Persian in the modern world of information technology. *Name-ye Farhangestan*, 8(3), 59-70 .[In Persian].
- Azmi, R., & Kabir, E. (2001). A new segmentation technique for omnifont Farsi text. *Pattern Recognition Letters, 22*(2), 97-104.
- Bonyani, M., & Jahangard, S. (2020). Persian handwritten digit, character, and words recognition by using deep learning methods. ArXiv e-prints, arXiv-2010, 1-14.
- Choudhary, A. (2014). A review of various character segmentation techniques for cursive handwritten words recognition. *International Journal of Information Computer Technology, 4*(6), 559–564.

- Ebrahimi, A., & Kabir, E. (2008). A pictorial dictionary for printed Farsi subwords. *Pattern Recognition Letters, 29*(5), 656–663.
- Hamad, K. A., & Kaya, M. (2016). A detailed analysis of optical character recognition technology. *International Journal of Applied Mathematics, Electronics and Computers, 4*(3), 244–249.
- Khosravi, H., & Kabir, E. (2009). A blackboard approach towards integrated Farsi OCR System. *International Journal of Document Analysis and Recognition*, *12*(1), 21–32.
- Khosrobeigi, Z., Veisi, H., Ahmadi, H.R., & Shabanian, H. (2020). A rule-based post-processing approach to improve Persian OCR performance. *Scientia Iranica D,* 27(6), 3019-3033.
- Kiaei, P., Javaheripi, M., & Mohammadzade, H. (2019). High accuracy Farsi language character segmentation and recognition. In 2019 27th *Iranian Conference on Electrical Engineering (ICEE),* 1692-1698.
- Margner, V., & El-Abed, H. (2008). Databases and competitions: strategies to improve Arabic recognition systems, Arabic and Chinese Handwriting Recognition. *Lecture Notes in Computer Science,* vol. 4768, Springer, 82–103.
- Maskanati, S., & Keshavarz, A. (2017). Online Persian hand writing recognition using language model and reduction of user writing rules". *Signal and Data Processing, 14*(2), 3-24.
- Menhaj, M.B., & Adab, M. (2002). Simultaneous segmentation and recognition of Farsi/Latin printed texts with MLP. In 2002 *International Joint Conference on Neural Networks* (1534–1539).
- Mozaffari, S., Faez, K., & Ziaratban, M. (2005). Structural decomposition and statistical description of farsi/arabic handwritten numeric characters. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (237–241).

- Parhami, B., & Taraghi, M. (1981). Automatic recognition of printed Farsi texts. *Pattern Recognition, 14(1-6), 395-403*.
- Pourreza, M., Derakhshan, R., Bibak, S., Fallah, M., Fayyazi, H., & Sabokrou, M. (2020). Persian OCR with cascaded convolutional neural networks supported by language model. In 2020 *10th International Conference on Computer and Knowledge Engineering*, (227-232).
- Rahmati, M., Fateh, M., Rezvani, M., Tajary, A., & Abolghasemi, V. (2020). Printed Persian OCR system using deep learning. *IET Image Processing, 14*(15), 3920-3931.
- Riaz, A. (2018). *An End-to-End OCR System for Pashto Cursive Script*. [Unpulished doctoral Dissertation]. Department of Computer Science of the University of Kaiserslautern, Germany.
- Shafii, M. (2014). *Optical Character Recognition of Printed Persian/Arabic Documents*. [Unpulished doctoral Dissertation]. University of Windsor, Canada.

[DOR: 20.1001.1.23223081.1401.0.0.114.6]

[DOI: 10.29252/LRR.14.2.11]