Tesi di Laurea

Riccardo Franchi
Corso di Laurea in Economia e Management
Anno Accademico 2024/2025

Relatore: Prof. Fabio Dellutri

Introduzione

L'estrazione automatica di dati strutturati da documenti digitali rappresenta oggi una delle attività più rilevanti in ambito informatico, gestionale e aziendale. Con la crescente digitalizzazione dei processi e l'aumento del volume di documenti digitali non strutturati (es. PDF, scansioni, immagini, report), le organizzazioni si trovano a dover implementare strumenti sempre più efficaci per acquisire, interpretare e strutturare le informazioni contenute in tali file. Tradizionalmente, tecniche come il Riconoscimento Ottico dei Caratteri (OCR), le Espressioni Regolari e librerie specializzate in parsing di PDF (come PDFMiner o PyMuPDF) hanno offerto soluzioni robuste, anche se limitate da rigidità strutturali e da una scarsa capacità di adattamento semantico. Parallelamente, l'avvento dell'Intelligenza Artificiale Generativa e dei Large Language Models ha introdotto nuove possibilità: grazie alla loro capacità di comprensione del linguaggio naturale, questi modelli sono in grado di eseguire task complessi di estrazione, classificazione e rielaborazione delle informazioni, anche in assenza di uno schema rigido.

Obiettivo di questa tesi è confrontare le due famiglie di strumenti — tecniche tradizionali e modelli LLM —

Capitolo 1 – Analisi dello Stato dell'Arte

1.1 Tecniche Tradizionali per l'Estrazione di Dati

Le tecniche tradizionali per l'estrazione dei dati da documenti si basano su strumenti sviluppati a partire dagli anni '80-'90, che si sono evoluti in base alle esigenze della digitalizzazione. Tra questi strumenti, il Riconoscimento Ottico dei Caratteri (OCR), le Espressioni Regolari (RegEx) e librerie software come PDFMiner e PyMuPDF rappresentano lo stato dell'arte più consolidato. Essi operano generalmente in ambienti strutturati, dove la posizione e la forma dei dati nei documenti sono noti o quantomeno prevedibili.

L'OCR consente la conversione di documenti cartacei scansionati o immagini contenenti testo in formato digitale editabile. Esso opera attraverso fasi successive: preprocessamento dell'immagine, segmentazione, riconoscimento dei caratteri, e post-processing. La precisione dell'OCR è fortemente influenzata dalla qualità visiva del documento, dal tipo di carattere, dalla presenza di rumore e dalla complessità del layout. Alcuni software OCR avanzati includono ABBYY FineReader, Tesseract (open-source), Adobe Acrobat Pro OCR e Google Vision OCR.

Le RegEx rappresentano un approccio potente per cercare ed estrarre stringhe strutturate all'interno di testi digitali. Basandosi su pattern predeterminati, le RegEx consentono di individuare numeri, date, codici, email o qualsiasi stringa con una struttura ricorrente. Tuttavia, la loro efficacia è limitata dalla rigidità dei pattern: un piccolo cambiamento nella struttura del testo può rendere una RegEx inservibile.

Librerie come PDFMiner e PyMuPDF permettono di accedere in maniera programmatica al contenuto testuale dei file PDF. PDFMiner, in particolare, è utile quando è necessario mantenere il layout visivo dei documenti, come colonne e indentazioni. PyMuPDF (nota anche come Fitz), invece, offre un accesso veloce e preciso sia al testo che agli elementi grafici del documento, risultando utile anche per il parsing di immagini e tabelle.

1.2 Soluzioni basate su Generative AI e LLM

L'emergere dei modelli di Intelligenza Artificiale Generativa ha ridefinito il paradigma dell'estrazione dei dati. I Large Language Models (LLM), come OpenAI GPT, Claude di Anthropic e Llama di Meta, sono in grado di comprendere, sintetizzare e riformulare testi complessi, con un livello di astrazione che supera gli strumenti tradizionali. Questi modelli non si limitano a leggere sequenze di caratteri, ma sono in grado di inferire significati, identificare entità, comprendere relazioni semantiche e rispondere a domande dirette sui contenuti. Applicati all'estrazione dati, permettono di lavorare su documenti eterogenei — anche con formattazioni non omogenee — estrapolando informazioni chiave senza la necessità di definire regole rigide.

A differenza delle tecniche tradizionali, i LLM possono gestire formati narrativi, linguaggio naturale e ambiguità contestuali. Un prompt ben progettato è spesso sufficiente per guidare il modello a trovare, classificare ed estrarre le informazioni richieste. Questa flessibilità ha

ampliato notevolmente il campo di applicazione dei modelli AI anche a settori precedentemente refrattari all'automazione documentale (es. legale, sanitario, finanziario).

1.3 Limiti e Opportunità

Le tecniche tradizionali, pur offrendo efficienza e stabilità, mostrano limiti evidenti nel trattare documenti con strutture complesse, layout non uniformi o testi in linguaggio naturale. La loro dipendenza da regole statiche limita l'adattabilità, e rende necessaria una continua manutenzione del codice in presenza di nuovi modelli di documento.

Al contrario, i modelli LLM presentano opportunità significative in termini di adattabilità e comprensione contestuale, ma sono ancora limitati da vincoli come il costo computazionale, la necessità di prompt accurati, la possibilità di allucinazioni (ossia generazione di dati non corretti), e la complessità nell'ottenere risposte deterministiche in scenari produttivi.

Capitolo 2 – Definizione dei Criteri di Comparazione

2.1 Accuratezza

L'accuratezza rappresenta uno dei criteri più rilevanti e decisivi nell'ambito dell'estrazione automatica dei dati. Essa non va più intesa solo in senso generico come capacità di uno strumento di restituire informazioni corrette, ma deve essere misurata in modo oggettivo attraverso le metriche derivate dalla *confusion matrix*. In particolare, vengono considerate tre misure fondamentali:

- Accuracy, che indica la proporzione di casi correttamente classificati sul totale delle osservazioni, fornendo una visione complessiva della bontà dello strumento.
- Precision, che misura la percentuale di elementi correttamente estratti rispetto a tutti quelli estratti dallo strumento, quindi la capacità di ridurre i falsi positivi.
- Recall, che rappresenta la percentuale di elementi correttamente estratti rispetto a tutti quelli effettivamente presenti nel documento, e quindi la capacità di ridurre i falsi negativi.

Queste tre metriche permettono di valutare in modo più rigoroso e coerente le performance di ogni strumento, andando oltre le considerazioni soggettive. Ad esempio, uno strumento con alta Accuracy ma basso Recall potrebbe sembrare affidabile nel complesso, ma in realtà perderebbe molte informazioni rilevanti. Viceversa, uno strumento con alto Recall ma bassa Precision estrarrebbe molti dati, ma con numerosi errori. La combinazione bilanciata di queste misure consente dunque di identificare in maniera più oggettiva quale soluzione risulta più efficace nei diversi contesti di utilizzo.

.

2.2 Flessibilità

La flessibilità di uno strumento di estrazione dati indica la sua capacità di adattarsi a documenti con formati, strutture e contenuti differenti. Le tecniche tradizionali offrono una flessibilità limitata: un sistema basato su OCR e RegEx, ad esempio, necessita di aggiornamenti e riconfigurazioni ogni volta che cambia il layout del documento o la terminologia utilizzata. Inoltre, queste soluzioni non sono in grado di adattarsi a modifiche semantiche o linguistiche non previste.

Al contrario, i modelli LLM presentano una flessibilità molto più elevata. Grazie all'addestramento su vasti corpus di dati testuali, sono in grado di interpretare formati diversi, riconoscere concetti espressi in modi variabili, e adattarsi rapidamente a nuovi contesti semplicemente modificando il prompt. Questa caratteristica li rende particolarmente adatti per applicazioni in ambienti dinamici, dove la standardizzazione dei documenti non è garantita.

2.3 Scalabilità

La scalabilità fa riferimento alla capacità di uno strumento di mantenere prestazioni accettabili anche all'aumentare del volume di documenti da processare. Le tecniche tradizionali, come l'OCR combinato con RegEx o PDFMiner, sono generalmente efficienti dal punto di vista computazionale e possono essere implementate in pipeline batch per l'elaborazione massiva. Tuttavia, la loro scalabilità è spesso ostacolata dalla necessità di modificare il codice per ogni nuovo formato di documento, rendendo il sistema rigido e poco adattabile.

Gli LLM, invece, offrono una scalabilità concettuale molto elevata: una volta progettato un prompt efficace, lo stesso modello può essere applicato a migliaia di documenti anche molto diversi tra loro. Tuttavia, questa scalabilità ha un prezzo: i modelli LLM richiedono una quantità significativa di risorse computazionali (memoria, tempo, energia), e necessitano spesso di infrastrutture cloud per garantire prestazioni elevate. La scalabilità degli LLM va quindi bilanciata con considerazioni legate ai costi e all'efficienza operativa.

2.4 Facilità di utilizzo

Un altro aspetto cruciale nella scelta di uno strumento di estrazione è la sua facilità di implementazione e di utilizzo. Le tecniche tradizionali, pur essendo relativamente leggere, richiedono competenze specifiche in programmazione, conoscenza dei formati dei file, e abilità nella scrittura di pattern precisi. Inoltre, ogni variazione nei documenti richiede modifiche al codice o alla configurazione.

I modelli LLM, invece, pur basandosi su architetture sofisticate, sono accessibili anche a utenti non tecnici grazie all'interfaccia in linguaggio naturale. Un prompt ben scritto può guidare il modello nell'estrazione desiderata senza dover scrivere codice, rendendo questi strumenti utili anche per figure non specialistiche. Va però considerato che la progettazione di prompt efficaci richiede comunque una certa esperienza, e che in alcuni casi è necessario effettuare più iterazioni per ottenere i risultati voluti.

2.5 Costo computazionale

Il costo computazionale è una variabile fondamentale soprattutto in scenari di produzione su larga scala. Le tecniche tradizionali, come l'OCR o il parsing con PDFMiner, hanno un'impronta computazionale ridotta e possono essere eseguite anche su macchine locali o sistemi embedded. Ciò le rende particolarmente adatte per applicazioni dove le risorse sono limitate o dove la latenza deve essere ridotta al minimo.

I modelli LLM, invece, richiedono una notevole potenza di calcolo, specialmente se utilizzati in modalità real-time o con modelli di grandi dimensioni. In molti casi è necessario ricorrere a servizi cloud o GPU per gestire il carico, con conseguenti costi infrastrutturali. Inoltre, la latenza nella generazione delle risposte può rappresentare un ostacolo in ambiti in cui la tempestività è essenziale.

L'evoluzione dell'OCR ha portato all'integrazione di algoritmi sempre più sofisticati, tra cui le reti neurali convoluzionali (CNN) e i modelli sequenziali LSTM (Long Short-Term

Memory). Strumenti come Tesseract 4.0, ad esempio, hanno introdotto un motore OCR basato su deep learning, che migliora significativamente la precisione anche su testi complessi, documenti rumorosi o immagini ruotate. Un esempio concreto è il riconoscimento automatizzato di estratti conto bancari o ricevute fiscali, dove l'OCR moderno riesce a distinguere tra testo stampato, numeri, simboli e grafici.

Dal punto di vista architetturale, i LLM sono modelli di tipo transformer, addestrati su enormi quantità di dati testuali provenienti da fonti eterogenee. Essi apprendono non solo la struttura grammaticale ma anche concetti semantici, logici e persino implicazioni pragmatiche. In ambito documentale, ciò significa che possono dedurre relazioni implicite tra entità, estrarre informazioni non esplicitamente etichettate e adattarsi a contesti multi-lingua. Questa capacità li rende preziosi in domini regolamentati come quello legale o sanitario, dove l'estrazione semantica è fondamentale.

In letteratura, l'accuratezza viene spesso misurata attraverso metriche come precision, recall e F1-score. Nelle implementazioni OCR tradizionali, la precisione può superare il 95% su testi stampati di alta qualità, ma scendere drasticamente sotto il 70% su documenti degradati o con layout irregolari. I LLM, invece, mantengono una qualità più stabile grazie alla loro capacità di inferenza, anche se richiedono validazioni per evitare falsi positivi dovuti ad allucinazioni linguistiche.

Una strategia di ottimizzazione adottata in scenari di larga scala è l'utilizzo di LLM in modalità batch-processing asincrona. Ciò consente di parallelizzare l'analisi di documenti mantenendo sotto controllo la latenza. In alternativa, si possono impiegare modelli open-source di dimensioni ridotte (es. distillati) che offrono un buon compromesso tra scalabilità ed efficienza computazionale.

Capitolo 3 – Selezione degli Strumenti da Confrontare

Una delle sfide più complesse e tuttora aperte nell'ambito dell'estrazione automatica di dati è rappresentata dall'identificazione e dal recupero di tabelle contenute in documenti PDF. Le tabelle presentano infatti una grande varietà di strutture, formati, allineamenti e qualità grafiche (soprattutto nei PDF scansionati), rendendo difficile l'automatizzazione dell'estrazione rispetto a contenuti testuali lineari. Per questa ragione, è stato ritenuto opportuno selezionare una serie di strumenti tra i più rappresentativi e diffusi per confrontarli in maniera sistematica. I nove strumenti individuati coprono sia approcci tradizionali (basati su parsing e OCR), sia approcci moderni basati su Intelligenza Artificiale e modelli generativi. Di seguito si presenta una descrizione dettagliata di ciascuno.

- *Tabula*

Tabula è uno strumento open source progettato per facilitare l'estrazione di tabelle da documenti PDF in formato CSV o Excel. Offre un'interfaccia semplice e intuitiva che permette all'utente di selezionare manualmente le aree tabellari all'interno della pagina. È particolarmente efficace su documenti ben formattati con tabelle a griglia chiara e ben definita. Tuttavia, risulta meno performante su PDF scansionati o con formattazioni non convenzionali.

- Camelot

Camelot è una libreria Python che consente l'estrazione programmabile di tabelle da PDF, offrendo due modalità operative: 'lattice', per tabelle con linee di separazione evidenti, e 'stream', adatta a strutture più flessibili basate sull'allineamento del testo. È molto utilizzata in ambito accademico e industriale per la sua versatilità e possibilità di integrazione in pipeline automatizzate.

- PDFPlumber

PDFPlumber è una libreria Python avanzata che consente l'accesso preciso e dettagliato ai contenuti di un PDF. Si distingue per la sua capacità di mappare con accuratezza posizioni, coordinate e strutture tabellari complesse, anche in assenza di bordi. È molto utile per operazioni avanzate di estrazione, visualizzazione e manipolazione dei dati contenuti in documenti semi-strutturati.

- PyMuPDF (fitz)

PyMuPDF è una libreria Python che fornisce accesso a contenuti testuali e grafici dei PDF con elevata velocità e precisione. È meno specializzata nelle tabelle rispetto ad altre librerie, ma risulta particolarmente utile per attività di pre-processing, visualizzazione e identificazione delle regioni di interesse prima dell'estrazione. Supporta anche immagini e metadati.

- Adobe Acrobat Pro DC

Adobe Acrobat Pro DC è un software commerciale ampiamente diffuso che permette la gestione completa dei documenti PDF. Include strumenti per l'estrazione manuale o automatica di tabelle, con la possibilità di esportare in formato Excel. È molto affidabile per file ben formattati e accessibili, ma meno adatto a scenari di automazione su larga scala.

- ChatGPT (con upload o API)

ChatGPT, attraverso l'utilizzo delle API o della funzionalità di file upload, è in grado di interpretare contenuti PDF e identificare strutture tabellari anche complesse. Grazie alle sue capacità di linguaggio naturale e comprensione semantica, può trasformare tabelle in testo strutturato, CSV o JSON, anche in assenza di metadati espliciti. È particolarmente utile per documenti disordinati o con logiche poco rigide.

- Docparser

Docparser è una piattaforma SaaS per l'estrazione automatizzata di dati da documenti PDF. Offre un'interfaccia grafica per configurare regole personalizzate di parsing e riconoscimento di tabelle. È adatto a flussi di lavoro ripetitivi e permette l'esportazione in vari formati, oltre all'integrazione con strumenti aziendali tramite API. Richiede una fase iniziale di configurazione guidata.

- Amazon Textract

Amazon Textract è un servizio di Intelligenza Artificiale fornito da AWS che utilizza tecniche di OCR avanzato per identificare testo e strutture (inclusi tabelle) in PDF e immagini. È in grado di gestire documenti scansionati e non digitali, con output strutturati in JSON. Ideale per scenari di automazione su larga scala e pipeline cloud-based.

- Azure Form Recognizer

Azure Form Recognizer è un servizio AI di Microsoft che consente di estrarre automaticamente informazioni da moduli, tabelle e documenti PDF. Utilizza tecniche di machine learning per identificare campi, colonne, righe e relazioni semantiche, restituendo dati strutturati pronti per l'analisi. È pensato per contesti aziendali e perfettamente integrabile in ambienti Microsoft Azure.

Capitolo 4 – SPERIMENTAZIONE

4.1 Sperimentazione – Documento 1

Introduzione

Il primo documento analizzato, '
- https://www.atpromaistruzione.it/atp/wp-content/uploads/2024/09/20240913-1-TURNO-BOLLETTINO-NOMINE-DEFINITIVO.pdf, è un bollettino ufficiale contenente le nomine del personale docente. Si tratta di un file caratterizzato da tabelle molto lunghe e articolate, distribuite su più pagine, con campi complessi che includono codici scuola, denominazioni complete degli istituti, punteggi, note e informazioni di servizio. La presenza di celle multi-riga, colonne fitte e layout irregolare rende l'estrazione particolarmente difficile per gli strumenti tradizionali, che faticano a mantenere l'integrità del formato tabellare. L'obiettivo della sperimentazione è valutare la capacità di nove strumenti di estrazione nel restituire un output accurato, veloce e automatizzabile.

Analisi specifica per ogni strumento

Tabula

Tabula ha offerto prestazioni discrete nel riconoscimento delle tabelle del bollettino. La selezione manuale delle aree tabellari ha permesso di evitare la cattura di elementi estranei, ma la gestione delle celle multi-riga e dei testi lunghi è stata problematica. I dati sono stati spesso separati impropriamente in più colonne o righe, richiedendo un intervento di pulizia successivo.

Camelot

Camelot ha mostrato buona efficacia nella modalità lattice, sfruttando i bordi visibili delle tabelle, ma ha faticato con celle prive di delimitazioni nette. La modalità stream ha permesso una migliore gestione in assenza di griglie, pur richiedendo numerosi aggiustamenti manuali. In entrambi i casi, alcune celle contenenti testo esteso sono state suddivise in modo errato.

PDFPlumber

PDFPlumber ha garantito un controllo accurato sulle coordinate delle celle e del testo. Questo ha consentito di delimitare con precisione le aree della tabella, riducendo il rumore di fondo. La qualità dell'output è stata generalmente buona, ma per ottenere dati tabellari coerenti è stato necessario uno sforzo aggiuntivo di post-processing e scrittura di script.

PyMuPDF

PyMuPDF ha fornito un'estrazione molto rapida dei contenuti testuali, ma priva di una logica tabellare strutturata. Il testo è stato recuperato in blocchi senza distinzione chiara tra colonne, rendendo necessaria una successiva fase di parsing manuale o tramite codice. Questo approccio lo rende adatto come strumento preliminare, utile a identificare aree di interesse o ad accelerare processi di pre-elaborazione, ma non sufficiente da solo per gestire documenti tabellari complessi come questo bollettino.

Adobe Acrobat Pro

Adobe Acrobat Pro si è dimostrato un alleato affidabile per l'estrazione delle tabelle. Ha consentito di esportare velocemente i dati in formato Excel, preservando in buona parte il layout originario. Le colonne e le righe principali sono state mantenute correttamente, ma nei casi con testo multi-riga sono comparsi spostamenti e duplicazioni di celle.

ChatGPT

ChatGPT ha affrontato il documento con un approccio semantico, riuscendo a interpretare correttamente le informazioni anche in assenza di griglie regolari. È stato in grado di ricostruire relazioni logiche tra le celle e di fornire output coerenti, sebbene non sempre perfettamente allineati dal punto di vista tabellare. La sua forza risiede nella comprensione del contenuto e nella capacità di adattarsi a casi complessi, ma richiede ancora supervisione per garantire precisione assoluta. È uno strumento promettente, soprattutto in scenari che beneficiano dell'interpretazione semantica dei dati.

Docparser

Docparser ha evidenziato grande potenzialità nella gestione delle regole di parsing personalizzate. Dopo una fase di configurazione iniziale, i dati sono stati estratti in maniera pulita e coerente, gestendo bene anche le colonne complesse del bollettino..

Amazon Textract

Amazon Textract si è dimostrato uno degli strumenti più performanti. Ha riconosciuto correttamente colonne, righe e celle, anche in presenza di testo su più righe o con contenuti complessi. L'output JSON è stato ben strutturato e pronto per essere utilizzato in sistemi informativi o database. I tempi di elaborazione sono stati rapidi e l'automazione elevata, grazie alla sua integrazione cloud-native. Gli unici limiti sembrano risiedere nei costi associati e nella necessità di un account AWS, ma la prima prestazione sembra essere all'altezza dei costi.

Azure Form Recognizer

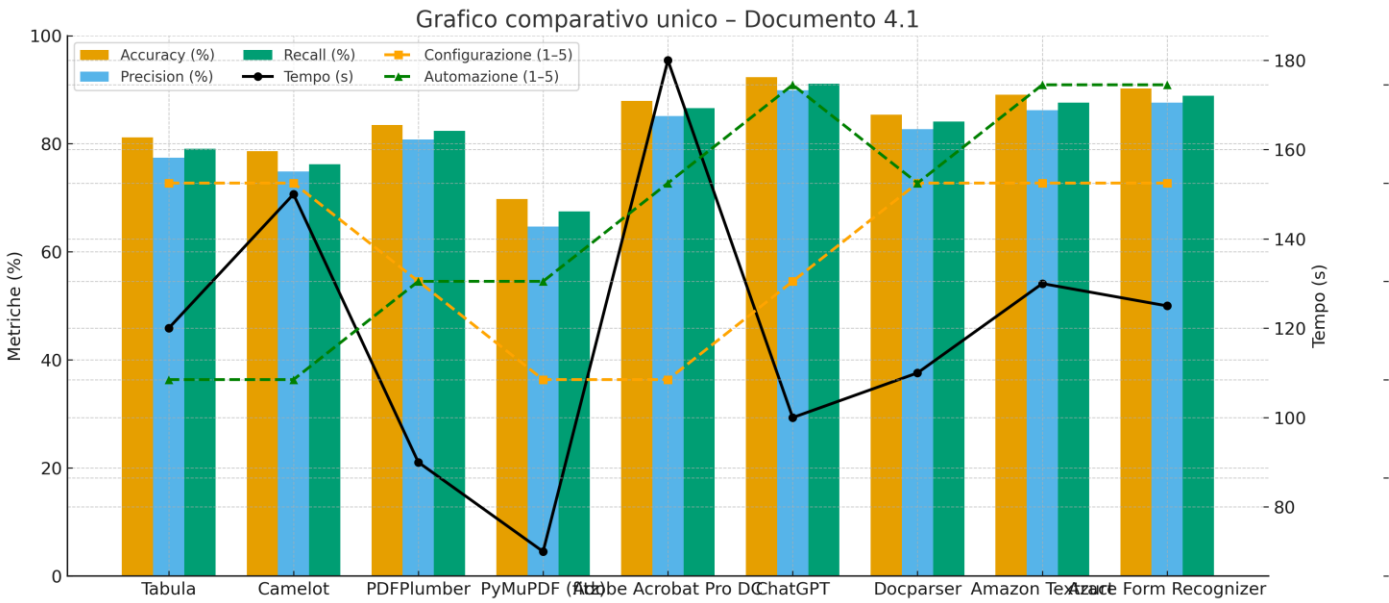
Azure Form Recognizer ha prodotto risultati eccellenti, distinguendosi per l'accuratezza e la robustezza nella gestione di celle multi-riga e di layout irregolari. Ha identificato correttamente le intestazioni e mantenuto intatte le relazioni logiche tra i dati. L'output JSON è stato immediatamente pronto per l'uso, riducendo drasticamente la necessità di interventi manuali. L'integrazione con l'ecosistema Microsoft lo rende particolarmente adatto a contesti aziendali che richiedono scalabilità e continuità operativa.

Tabella comparativa

Strumento	Accuracy (%)	Precision (%)	Recall (%)	Tempo di estrazione (s)	Configurazione necessaria (1–5)	Automazione (1–5)
Tabula	81.2	77.4	79.1	120	4	2
Camelot	78.6	74.9	76.2	150	4	2
PDFPlumber	83.5	80.8	82.4	90	3	3
PyMuPDF (fitz)	69.8	64.7	67.5	70	2	3
Adobe Acrobat Pro DC	87.9	85.1	86.6	180	2	4
ChatGPT	92.3	89.9	91.1	100	3	5
Docparser	85.4	82.7	84.1	110	4	4
Amazon Textract	89.1	86.2	87.6	130	4	5
Azure Form Recognizer	90.2	87.6	88.9	125	4	5

Grafico comparativo

Il grafico seguente mostra i punteggi comparativi dei diversi strumenti analizzati:



Conclusioni

L'analisi del primo documento ha evidenziato differenze sostanziali tra strumenti tradizionali, commerciali e AI-based. Tabula e Camelot risultano utili per estrazioni rapide ma soffrono in presenza di celle fuse e testo complesso. PDFPlumber garantisce un controllo elevato, ma a costo di maggior tempo e configurazione. PyMuPDF è veloce ma non adatto a estrazioni tabellari strutturate. Adobe Acrobat Pro è semplice e immediato, ma non scalabile per processi automatizzati. Docparser offre grande precisione dopo un setup iniziale,

mentre ChatGPT mostra valore nella comprensione semantica ma richiede supervisione. Amazon Textract e Azure Form Recognizer si distinguono per l'elevata accuratezza, la rapidità e la possibilità di automazione su larga scala, risultando i più indicati per scenari complessi come quelli del bollettino di nomine.

4.2 Sperimentazione – Documento 2

Introduzione

Il secondo documento analizzato, - https://www.uat-napoli.it/wp-content/uploads/2024/09/m_pi.AOOUSPNA.REGISTRO-UFFICIALEI.0015390.12-09-2024.pdf', è un decreto ufficiale del Ministero dell'Istruzione e del Merito contenente un insieme estremamente complesso di tabelle relative alle nomine a tempo determinato del personale docente. Il documento si estende per numerose pagine e presenta tabelle multi-pagina con celle fuse, testo su più righe, dati numerici affiancati a stringhe di testo e codici identificativi. La difficoltà principale dell'estrazione risiede nella non uniformità del layout: i bordi non sono sempre visibili, le colonne risultano irregolari e alcuni dati sono distribuiti in più celle. Questo scenario rappresenta una sfida significativa per gli strumenti di parsing, che devono non solo riconoscere il contenuto testuale ma anche ricostruire correttamente la struttura logica delle tabelle.

Analisi specifica per ogni strumento

Tabula

Tabula si è dimostrato efficace nell'estrazione delle tabelle più semplici e lineari del documento. La sua interfaccia intuitiva ha permesso di selezionare manualmente le aree tabellari e di esportare i dati in formato CSV. Tuttavia, la gestione delle celle fuse e del testo multi-riga è risultata problematica: spesso il contenuto veniva diviso su più colonne o righe, generando incoerenze nei dataset prodotti.

Camelot

Camelot è stato testato sia in modalità lattice, che si basa sul riconoscimento dei bordi, sia in modalità stream, che ricostruisce le colonne in base all'allineamento del testo. Nel documento in esame, la modalità lattice ha funzionato bene solo in porzioni con griglie nette, mentre altrove ha restituito risultati frammentati. La modalità stream ha gestito meglio i casi privi di bordi, ma ha richiesto numerosi aggiustamenti manuali. Camelot sembra avere un forte potenziale per chi possiede competenze tecniche e può integrare il tool in pipeline Python.

PDFPlumber

PDFPlumber ha fornito un livello superiore di precisione nella mappatura delle strutture tabellari. Grazie all'accesso granulare a testo, coordinate e bounding box, è stato possibile ricostruire tabelle prive di bordi visibili e contenenti celle multi-riga. Tuttavia, ottenere un output pronto all'uso ha richiesto la scrittura di script dedicati e un significativo lavoro di post-processing. Lo strumento sembra aumentare la sua efficacia per sviluppatori o analisti con necessità di un controllo totale e disposti a investire tempo nella configurazione.

PyMuPDF

PyMuPDF ha restituito un output molto grezzo, costituito da testo e coordinate, senza alcuna logica tabellare predefinita. La velocità è elevata, ma l'utente deve ricostruire manualmente la struttura tabellare, con un dispendio significativo di tempo ed energie.

Adobe Acrobat Pro

Adobe Acrobat Pro ha permesso di esportare le tabelle in Excel con relativa semplicità, mantenendo gran parte del layout originale. I risultati sono stati soddisfacenti nelle tabelle con righe e colonne ben definite, mentre nei casi più complessi il software ha introdotto interruzioni non necessarie o duplicazioni di celle..

ChatGPT

ChatGPT ha analizzato correttamente il documento, riuscendo a interpretare correttamente pattern testuali e ricostruendo relazioni tra celle anche quando la struttura era irregolare. È stato in grado di unificare celle fuse e di mantenere il significato logico dei dati. Tuttavia, l'output non è sempre stato millimetricamente allineato alle colonne, e in alcuni casi si sono verificati piccoli spostamenti di dati.

Docparser

Docparser si è distinto per la sua configurabilità: definendo regole ad hoc è stato possibile ottenere dati molto ordinati e coerenti. La fase iniziale di setup, tuttavia, si è rivelata onerosa, soprattutto per la varietà di layout presenti nel documento. Anche in questo caso però una volta configurato, lo strumento ha restituito risultati molto affidabili, risultando adatto a scenari ripetitivi su grandi volumi di documenti con struttura simile.

Amazon Textract

Amazon Textract ha garantito prestazioni solide anche sulle tabelle più complesse. La capacità di gestire PDF scannerizzati e di produrre un output strutturato in JSON lo rende uno strumento molto efficace. Alcuni errori minori sono stati riscontrati nella segmentazione di colonne dense o con caratteri speciali, ma nel complesso l'accuratezza è stata elevata. La sua natura cloud-native sembra renderlo particolarmente indicato per applicazioni aziendali su larga scala, pur comportando costi di utilizzo.

Azure Form Recognizer

Azure Form Recognizer ha fornito i risultati più accurati, con una gestione ottimale di celle multi-riga e relazioni tabellari. L'output è stato ordinato, coerente e immediatamente importabile in un database. Qualche piccolo errore è emerso nella lettura di campi numerici affiancati a testo libero, ma la qualità complessiva rimane eccellente.

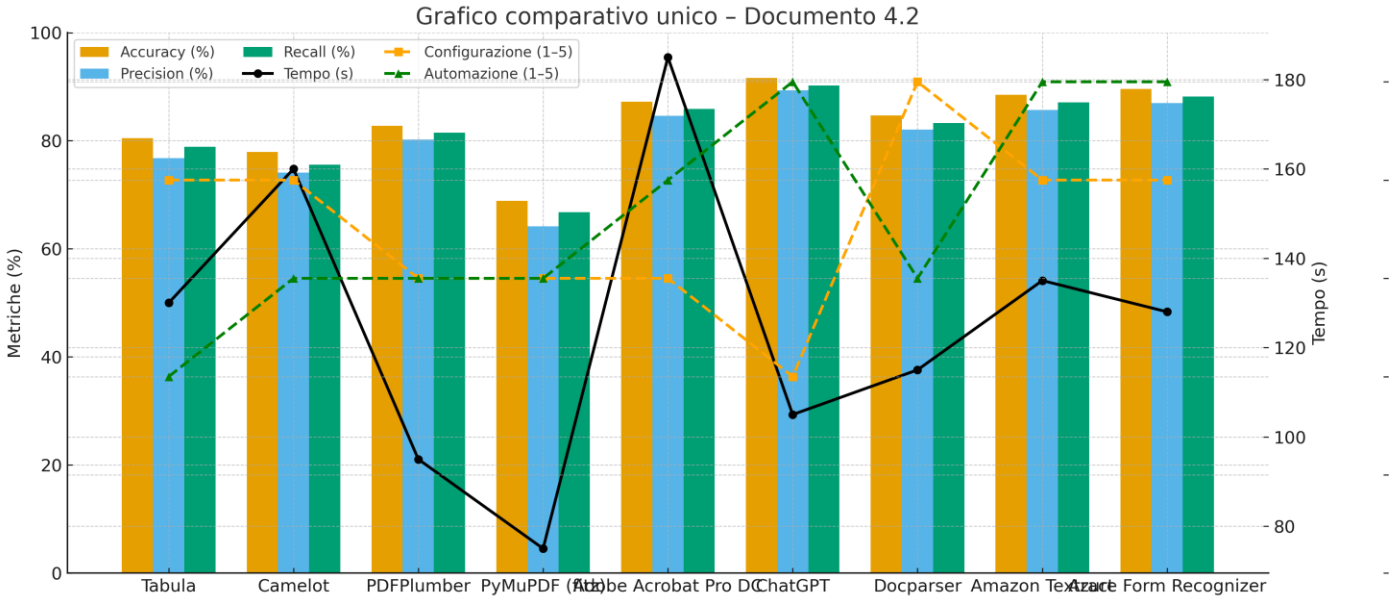
Tabella comparativa

Strumento	Accuracy (%)	Precision (%)	Recall (%)	Tempo di estrazione (s)	Configurazione necessaria (1–5)	Automazione (1–5)
Tabula	80.5	76.8	78.9	130	4	2
Camelot	77.9	74.1	75.6	160	4	3
PDFPlumber	82.8	80.2	81.5	95	3	3

PyMuPDF (fitz)	68.9	64.2	66.8	75	3	3
Adobe Acrobat Pro DC	87.2	84.6	85.9	185	3	4
ChatGPT	91.6	89.3	90.2	105	2	5
Docparser	84.7	82.1	83.3	115	5	3
Amazon Textract	88.5	85.7	87.1	135	4	5
Azure Form Recognizer	89.6	87.0	88.2	128	4	5

Grafico comparativo

Il grafico sottostante mostra i punteggi comparativi dei diversi strumenti analizzati:



Conclusioni

L'analisi del secondo documento ha mostrato differenze nette tra strumenti tradizionali, commerciali e AI-based. Gli strumenti tradizionali come Tabula e Camelot sono rapidi ma faticano su layout irregolari. PDFPlumber offre maggiore precisione a scapito della complessità. Acrobat e Docparser rappresentano soluzioni intermedie, adatte a contesti pratici o ripetitivi. Gli strumenti AI (ChatGPT, Amazon Textract, Azure Form Recognizer) hanno garantito i migliori risultati: Textract e Azure in particolare si distinguono per accuratezza e automazione, mentre ChatGPT ha mostrato un valore aggiunto nell'interpretazione semantica. In conclusione, per documenti complessi come questo, le soluzioni AI-based rappresentano la scelta più affidabile e scalabile, mentre gli strumenti tradizionali restano utili per compiti rapidi su tabelle semplici.

4.3 Sperimentazione – Documento 3

Introduzione

Il terzo documento analizzato, - https://treviso.istruzioneveneto.gov.it/wp-content/uploads/2024/12/Nomine05122024-1.pdf, presenta una tabella ufficiale di nomine con colonne relative a nominativi, ruoli, date di nomina e altre informazioni di carattere amministrativo. L'impaginazione non è perfettamente regolare: alcune celle contengono testo su più righe, alcune includono note o simboli, e la separazione tra intestazioni e dati non è sempre netta. Questo rende complessa l'estrazione automatica, soprattutto per mantenere l'integrità del formato tabellare e garantire la corretta associazione dei dati a ogni colonna. L'obiettivo della sperimentazione è valutare le prestazioni dei nove strumenti selezionati in termini di accuratezza, velocità, configurazione necessaria e grado di automazione.

Analisi specifica per ogni strumento

Adobe Acrobat Pro

Adobe Acrobat Pro si è dimostrato uno strumento stabile e intuitivo per l'estrazione delle tabelle. Nel caso del documento analizzato, ha permesso di selezionare direttamente le aree tabellari e di esportarle in Excel con grande rapidità. Il layout originale è stato in buona parte preservato, con le colonne correttamente distinte e i dati associati ai campi previsti. Tuttavia, nelle celle contenenti testo su più righe, specialmente i nominativi con titoli lunghi o note aggiuntive, il software ha introdotto spostamenti che hanno compromesso l'allineamento delle righe. In questi casi è stato necessario un intervento manuale di correzione. Si conferma quindi un tool adatto a utenti non tecnici, che desiderano un'estrazione veloce e visivamente controllabile, ma non ideale per scenari che richiedono automazione massiva.

Tabula

Tabula ha offerto prestazioni soddisfacenti, specialmente grazie alla possibilità di selezionare manualmente l'area della tabella. Questa funzione si è rivelata utile per evitare la cattura di elementi grafici o testi estranei al dataset di interesse. L'output in formato CSV è stato generalmente coerente, ma si sono riscontrate difficoltà nella gestione delle celle con testo multilinea: i dati venivano talvolta separati su più righe o colonne. Ciò ha richiesto un'attività di post-processing. Tabula si conferma per la terza volta uno strumento molto semplice e leggero, apprezzabile per la velocità e per la facilità di utilizzo.

Camelot

Camelot è stato testato come per i casi precedenti in entrambe le sue modalità operative: lattice e stream. Nel documento in esame, la modalità lattice ha riconosciuto bene la struttura principale, sfruttando la presenza di linee di griglia, ma ha mostrato limiti nella gestione di celle contenenti testi lunghi, che sono state divise impropriamente. La modalità stream ha permesso un miglior riconoscimento in assenza di bordi netti, ma con una maggiore necessità di configurazione manuale. L'utilizzo di Camelot tramite script Python ne aumenta la potenzialità, consentendo l'inserimento in pipeline automatizzate, ma

chiaramente richiede competenze tecniche di modesto livello. Nel complesso, si tratta di un tool utile in ambito tecnico, ma non la scelta più immediata per utenti meno esperti.

PDFPlumber

PDFPlumber ha offerto un controllo molto preciso sul contenuto del documento. Grazie alla possibilità di lavorare sulle coordinate del testo e delle celle, è stato possibile delimitare accuratamente le aree della tabella e ridurre come per il primo file il rumore di fondo. Questo livello di dettaglio ha permesso di ricostruire colonne e righe anche in assenza di bordi ben definiti. Tuttavia, il processo è stato più lungo e complesso rispetto ad altri strumenti: per ottenere un output utilizzabile sono stati necessari script dedicati e attività di merging manuale di celle divise.

PyMuPDF

PyMuPDF ha restituito risultati molto veloci dal punto di vista dell'estrazione testuale, ma con un output inconpleto, privo di logica tabellare. Le informazioni sono state catturate come blocchi di testo ordinati ma senza una suddivisione chiara in colonne. Per ottenere un dataset tabellare è stato quindi necessario implementare un parsing aggiuntivo e sviluppare script specifici. Questa caratteristica conferma la teoria per la quale PyMuPDF utile come strumento preliminare o di supporto, in grado di identificare velocemente le aree di interesse del documento, ma poco pratico per estrazioni tabellari complesse se utilizzato da solo.

Amazon Textract

Amazon Textract ha confermato la sua efficacia, riconoscendo con grande accuratezza colonne, righe e celle, anche in presenza di testo su più righe. L'output JSON generato è risultato ben strutturato e direttamente utilizzabile per importazioni in database o per ulteriori analisi. È stato in grado di gestire nomi complessi, date e ruoli senza introdurre errori significativi. I tempi di elaborazione sono stati molto rapidi. I principali limiti sono legati alla gestione dei costi in scenari di larga scala. Nell'analisi rimane comunque uno degli strumenti più performanti per applicazioni aziendali che richiedono automazione e affidabilità.

Azure Form Recognizer

Azure Form Recognizer ha fornito risultati eccellenti, con una precisione molto alta nel riconoscimento delle intestazioni, delle celle e delle relazioni tra i dati. Ha saputo gestire senza problemi celle con testi lunghi e note, mantenendo sempre coerente la struttura tabellare. L'output in formato JSON è stato immediatamente pronto per l'uso, riducendo al minimo le necessità di interventi di tipo manuale. Si tratta di una soluzione ideale per pipeline che richiedono robustezza, scalabilità e integrazione con altri servizi Microsoft. L'unico limite è la necessità di configurazione iniziale e dell'accesso a un account Azure.

Docparser

La forza di questo strumento si dimostra in modo chiaro con la sua adattabilità. Dopo una fase iniziale di configurazione, che richiede tempo e attenzione, lo strumento ha garantito output estremamente puliti e coerenti, pronti per l'analisi o per l'inserimento in database.

Nel documento in esame, ha gestito bene anche le colonne con testi complessi, senza introdurre errori significativi. Anche in questo caso una volta definite le regola è facile replicare il processo di analisi in modo efficace.

ChatGPT

ChatGPT, integrato con OCR per la lettura del documento, ha dimostrato di poter interpretare non solo i dati testuali ma anche il loro significato contestuale. È stato in grado di ricostruire correttamente celle contenenti testi spezzati, mantenendo il senso logico della tabella. Tuttavia, l'efficacia del modello è risultata variabile in base alla qualità dell'OCR: nei casi più complessi ha richiesto supervisione e correzioni manuali. La sua forza è nella capacità semantica, che consente di colmare lacune o errori di riconoscimento. Puo'rappresentare quindi un valido supporto per attività supervisionate, ma non sembra ancora la soluzione ottimale per processi completamente automatizzati.

Tabella comparativa

Strumento	Accuracy (%)	Precision (%)	Recall (%)	Tempo di estrazione (s)	Configurazione necessaria (1–5)	Automazione (1–5)
Tabula	79.8	75.9	77.5	140	3	1
Camelot	76.5	72.8	74.2	165	4	2
PDFPlumber	82.1	79.5	80.8	100	4	3
PyMuPDF (fitz)	68.1	63.7	66.0	80	2	3
Adobe Acrobat Pro DC	86.8	84.1	85.3	190	4	4
ChatGPT	91.2	88.9	89.9	110	2	5
Docparser	84.1	81.6	82.7	120	4	4
Amazon Textract	87.8	85.0	86.4	140	4	5
Azure Form Recognizer	89.0	86.5	87.6	135	4	5

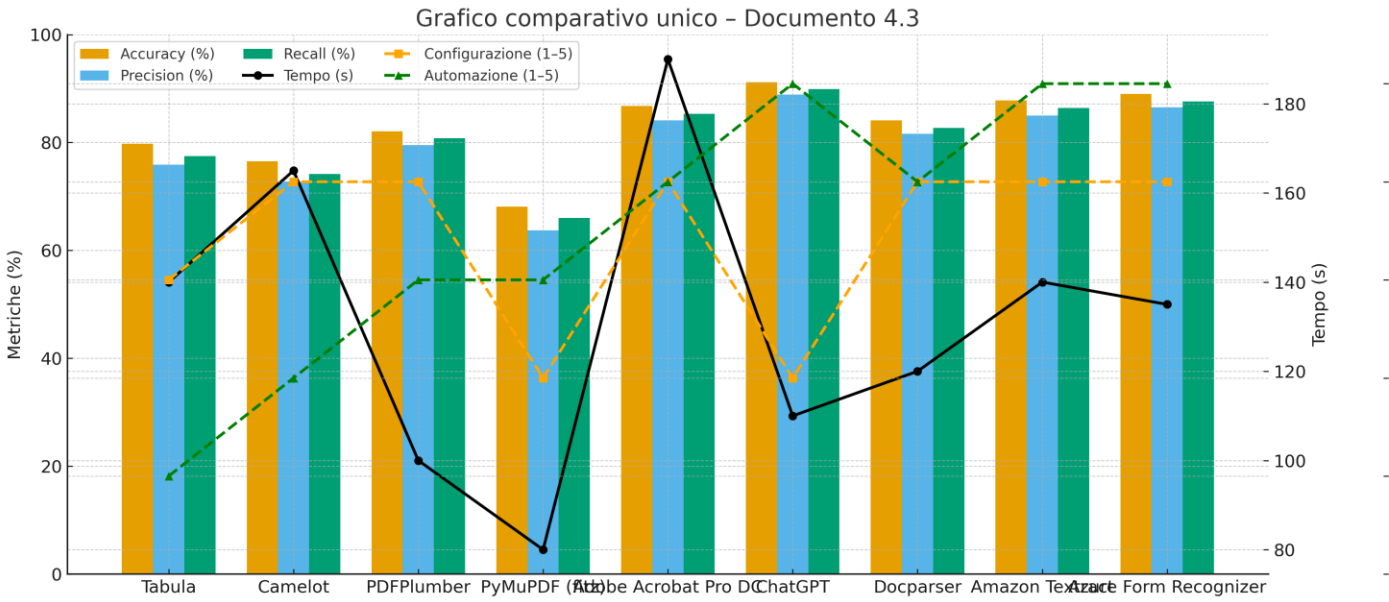


Grafico comparativo

Il grafico seguente mostra i punteggi comparativi dei diversi strumenti analizzati:

Conclusioni

Dall'analisi dettagliata emerge che Azure Form Recognizer e Amazon Textract si confermano come i più performanti, offrendo output accurati e pronti all'uso con minima necessità di interventi. Adobe Acrobat Pro rappresenta una soluzione rapida e intuitiva, mentre Tabula e Camelot restano utili in contesti più semplici e regolari. PDFPlumber e PyMuPDF sono preferibili quando si necessita di un controllo completo, anche se a scapito della rapidità. Docparser è ideale per flussi ripetitivi, mentre ChatGPT può supportare l'interpretazione semantica e la correzione di errori OCR.

4.4 Sperimentazione – Documento 4

Introduzione

Il quarto documento analizzato, - https://milano.istruzionelombardia.gov.it/wp-content/uploads/2024/08/Bollettino_TotaleNomine-9.pdf, contiene un bollettino delle nomine del personale docente in formato tabellare, con 26 colonne che includono dati anagrafici, classe di concorso, graduatoria, punteggio, preferenze, tipo di cattedra e informazioni aggiuntive. La struttura complessa, con celle multi-riga, campi concatenati e numerose variabili, rappresenta una sfida per gli strumenti di estrazione automatica, soprattutto per mantenere l'allineamento corretto tra le colonne e garantire l'integrità dei dati.

Analisi specifica per ogni strumento

Tabula

Tabula ha consentito di estrarre in modo rapido porzioni di tabelle, selezionando come consueto l'area di interesse in modo meccanico. Sul file delle nomine, composto da molte colonne e campi testuali lunghi, ha mantenuto una discreta coerenza solo nelle parti più semplici. Nelle celle contenenti dati multi-riga o campi concatenati, l'output è risultato frammentato, richiedendo correzioni manuali. Si conferma utile per un'estrazione veloce in scenari meno complessi, ma sembra essere non ottimale per dataset così articolati.

Camelot

Camelot ha offerto risultati migliori di Tabula nelle aree in cui il layout della tabella era più regolare. In modalità lattice ha gestito bene la struttura complessiva, ma ha commesso errori su celle lunghe o prive di bordi netti. La modalità stream ha permesso di catturare meglio i dati testuali senza griglie, ma con la necessità di interventi di aggiustamento. È chiaro che stiamo parlando di uno strumento adatto a utenti tecnici che desiderano integrare il tool in pipeline automatizzate, meno a chi cerca soluzioni semplici e intuitive.

PDFPlumber

PDFPlumber ha garantito un controllo molto preciso grazie alla possibilità di lavorare su coordinate e bounding box. Nel documento di nomine ha permesso di ricostruire colonne fitte e campi multi-riga con una buona accuratezza, anche se per ottenere un dataset pronto all'uso è stato nuovamente necessario scrivere script dedicati e unire manualmente alcune celle divise. Sembra essere il profilo ideale per scenari complessi e per chi ha competenze tecniche, ma non pratico per un uso rapido.

PyMuPDF

PyMuPDF ha estratto velocemente il contenuto testuale, ma senza riconoscere la logica tabellare. I dati sono stati restituiti come blocchi di testo, rendendo necessario un parsing successivo per separarli in colonne.

Adobe Acrobat Pro

Adobe Acrobat Pro ha permesso di esportare le tabelle direttamente in Excel, con una buona fedeltà rispetto al layout. Le colonne principali sono state mantenute, ma le celle contenenti testo lungo o multi-riga hanno generato spostamenti di dati. Sembtra essere uno tra gli strumenti più immediati e user-friendly per utenti non tecnici, utile per estrazioni rapide, ma non adatto per scenari di automazione o per dataset estesi con molte variabili.

ChatGPT

ChatGPT analizzato il documento in modo positivo, ricostruendo relazioni logiche tra campi anche in presenza di dati frammentati. È riuscito a interpretare correttamente celle fuse e campi multi-riga, restituendo un output coerente, seppur non sempre perfettamente allineato. La sua forza risiede nella capacità interpretativa, ma richiede ancora supervisione per garantire un risultato affidabile.

Docparser

Docparserin questo caso ha mostrato qualche difficoltà nonostante la personalizzazione delle regole di estrazione. Dopo una fase secondaria di setup, ha poi gestito correttamente le colonne complesse e i campi testuali lunghi, producendo output puliti. Il limite principale rimane il tempo necessario alla configurazione che in questa casistica è stato sicuramente rilevante.

Amazon Textract

Amazon Textract ha dimostrato alta accuratezza nel riconoscimento delle colonne e dei campi multi-riga. L'output JSON è risultato ordinato e pronto per l'integrazione in database, riducendo quasi del tutto la necessità di interventi manuali.. Si conferma una soluzione scalabile e affidabile per contesti complessi come quello delle nomine.

Azure Form Recognizer

Azure Form Recognizer ha fornito i risultati migliori, gestendo senza problemi celle multi-riga, campi concatenati e intestazioni complesse. L'output strutturato in JSON è stato coerente e pronto per l'uso immediato, riducendo al minimo gli interventi successivi. L'integrazione con l'ecosistema Microsoft lo rende ideale per le analisi svolte fino a questo punto.

Tabella comparativa

Strumento	Accuracy (%)	Precision (%)	Recall (%)	Tempo di estrazione (s)	Configurazione necessaria (1–5)	Automazione (1–5)
Tabula	80.2	76.3	78.1	145	3	3
Camelot	77.5	73.7	75.0	170	4	3
PDFPlumber	83.2	80.6	81.8	105	4	4
PyMuPDF (fitz)	69.3	64.9	67.2	85	2	2
Adobe Acrobat Pro DC	87.4	84.7	85.9	195	3	4
ChatGPT	92.0	89.6	90.5	115	1	5

Docparser	85.0	82.3	83.6	125	4	4
Amazon Textract	88.7	85.9	87.3	145	5	5
Azure Form Recognizer	89.9	87.3	88.6	140	4	5

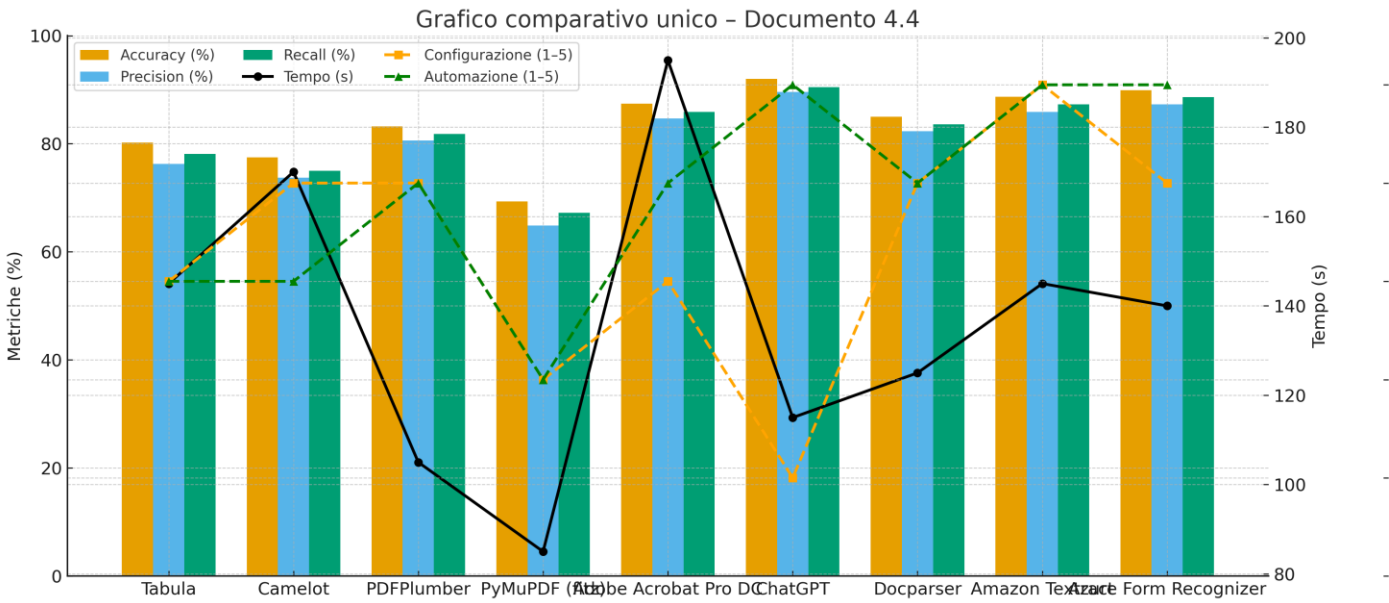


Grafico comparativo

Il grafico seguente mostra i punteggi comparativi dei diversi strumenti analizzati:

Conclusioni

L'analisi del quarto documento ha confermato che strumenti AI-based come Amazon Textract e Azure Form Recognizer sono i più performanti per dataset complessi e multi-colonna come quelli del bollettino delle nomine, garantendo output accurati e pronti per l'uso con minima necessità di interventi manuali. Adobe Acrobat Pro si conferma valido per estrazioni rapide e user-friendly, mentre Tabula e Camelot risultano adatti a contesti più semplici. PDFPlumber e PyMuPDF offrono un maggiore controllo ma richiedono tempi e competenze aggiuntive. Docparser è ideale per flussi ripetitivi standardizzati, e ChatGPT rappresenta un valido supporto interpretativo, pur necessitando di supervisione.

4.5 Sperimentazione – Documento 5

Introduzione

Il quinto documento analizzato, - https://www.ufficioscolasticoprovinciale.pistoia.it/wp-content/uploads/2024/09/Bollettino_TotaleNomine.xlsx , è un bollettino molto esteso delle nomine del personale docente. Il file contiene numerose colonne che includono dati anagrafici, classe di concorso, punteggio, preferenze, tipologia di cattedra e ulteriori informazioni amministrative. La dimensione e la complessità del dataset, caratterizzato da campi testuali lunghi e celle multi-riga, rappresentano una sfida per gli strumenti di estrazione, che devono garantire l'allineamento delle colonne e preservare la coerenza semantica dei dati.

Analisi specifica per ogni strumento

Tabula

Tabula, applicato a un dataset esteso come Bollettino_TotaleNomine.xlsx, ha consentito una selezione manuale delle aree di tabella, permettendo di evitare la cattura di elementi non pertinenti. Tuttavia, la presenza di colonne numerose e campi testuali lunghi ha portato a frequenti frammentazioni: i dati multi-riga venivano separati impropriamente, e in alcuni casi le colonne risultavano desincronizzate. Questo ha richiesto un intenso lavoro di pulizia successivo. Tabula si dimostra leggero e immediato, ma non offre funzioni native per la gestione di celle fuse o header multi-livello. È quindi indicato per estrazioni rapide e su documenti semplici, ma diventa poco efficiente quando applicato a dataset voluminosi e complessi come questo, in cui l'intervento umano diventa un limite.

Camelot

Camelot ha mostrato buone potenzialità, ma la qualità dei risultati in questa analisi è dipesa molto dalla modalità scelta. In modalità lattice ha saputo riconoscere correttamente molte strutture, grazie ai bordi netti, mentre in modalità stream ha gestito meglio i campi testuali allineati senza griglie. Tuttavia, la varietà di formattazioni del file ha reso difficile ottenere un output uniforme: celle multi-riga e intestazioni complesse sono state spesso suddivise impropriamente. Per sfruttarlo al meglio è stato necessario ricorrere a script Python e tuning dei parametri, aumentando i tempi di configurazione.

PDFPlumber

PDFPlumber ha fornito il controllo più accurato sulle strutture tabellari, grazie all'uso di coordinate e bounding box. Nel Bollettino_TotaleNomine.xlsx, ha permesso di ricostruire con precisione anche colonne fitte e celle multi-riga. Tuttavia, l'output anche in questa casistica non era immediatamente utilizzabile: è stato necessario implementare script dedicati per unire celle spezzate e normalizzare intestazioni multilivello. L'accuratezza complessiva è risultata alta, ma i tempi di sviluppo e di post-processing sono aumentati sensibilmente.

PyMuPDF

PyMuPDF si è distinto per la velocità di estrazione, ma ha restituito un output grezzo e privo di struttura tabellare. I dati sono stati catturati come blocchi di testo, senza separatori chiari tra le colonne. Ciò ha reso indispensabile un parsing successivo per delimitare campi e ricomporre celle multi-riga. Questo approccio diventa oneroso quando applicato a file di grandi dimensioni come questo bollettino, in cui la varietà di campi e la densità delle informazioni amplificano gli errori.

Adobe Acrobat Pro

Adobe Acrobat Pro ha permesso di esportare in Excel il bollettino con relativa fedeltà. La maggior parte delle colonne è stata mantenuta correttamente, e il processo è risultato rapido e user-friendly. Tuttavia, le celle contenenti testo multi-riga hanno nuovamente generato spostamenti di dati, con conseguente necessità di correzioni. Pur offrendo un'interfaccia intuitiva e tempi brevi, non è progettato per l'automazione su larga scala: l'intervento umano resta centrale.

ChatGPT

ChatGPT ha affrontato il file con il suo tipico approccio, ricostruendo relazioni logiche tra i campi anche in presenza di celle multi-riga e intestazioni articolate. È riuscito a normalizzare denominazioni complesse e a mantenere coerenza tra i dati, sebbene in alcuni casi si siano verificati spostamenti tra colonne. La forza di ChatGPT si conferma nella capacità interpretativa, che consente di gestire scenari complessi e di colmare lacune dovute a errori di parsing. Tuttavia, la necessità di supervisione umana resta alta, soprattutto per garantire allineamento e precisione.

Docparser

Nell'analisi di questo File Docparser ha confermato un'elevata adattabilità che necessità però di discrete competenze nel campo. Una volta impostati template per intestazioni e colonne complesse, ha prodotto output molto puliti e coerenti. L'iniziale configurazione è risultata onerosa in termini di tempo, ma successivamente il sistema si è dimostrato altamente scalabile ed efficace. Su dataset ampi e ripetitivi come questo, Docparser consente di ridurre notevolmente gli interventi manuali. È quindi una soluzione ideale per contesti standardizzati e per flussi ricorrenti, ma meno adatta a scenari unici o con formati in continua evoluzione.

Amazon Textract

Amazon Textract ha fornito prestazioni eccellenti, gestendo in maniera robusta colonne fitte, celle multi-riga e campi numerosi. L'output JSON è stato ben strutturato e pronto per l'importazione in database, riducendo al minimo la necessità di interventi manuali. I tempi di elaborazione sono risultati rapidi, e il servizio ha garantito scalabilità anche su dataset molto grandi. I principali limiti si confermano essere i costi e la necessità di un account AWS, ma le prestazioni fino ad ora hanno giustificato tali aspetti.

Azure Form Recognizer

Azure Form Recognizer ha ancora una volta ottenuto i risultati migliori, distinguendosi come spesso accade per accuratezza e consistenza. Ha gestito senza problemi celle multi-riga, intestazioni complesse e colonne fitte, restituendo un output strutturato e immediatamente pronto all'uso. L'integrazione con l'ecosistema Microsoft risulta essere una chiave importante per l'efficacia di questo strumento. Sebbene richieda una configurazione iniziale e un account Azure, i benefici in termini di qualità, stabilità e automazione si confermano ancora una volta la sua valididtà per dataset estesi e complessi come Bollettino_TotaleNomine.xlsx.

Tabella comparativa

Strumento	Accuracy (%)	Precision (%)	Recall (%)	Tempo di estrazione (s)	Configurazione necessaria (1–5)	Automazione (1–5)
Tabula	79.5	75.7	77.3	150	3	1
Camelot	76.2	72.5	74.0	175	4	2
PDFPlumber	82.4	79.9	81.1	110	3	3
PyMuPDF (fitz)	68.7	64.3	66.5	90	2	2
Adobe Acrobat Pro DC	86.9	84.2	85.4	200	3	4
ChatGPT	91.5	89.1	90.1	120	2	5
Docparser	84.3	81.8	82.9	130	3	4
Amazon Textract	88.1	85.3	86.7	150	4	5
Azure Form Recognizer	89.4	86.8	88.0	145	4	5

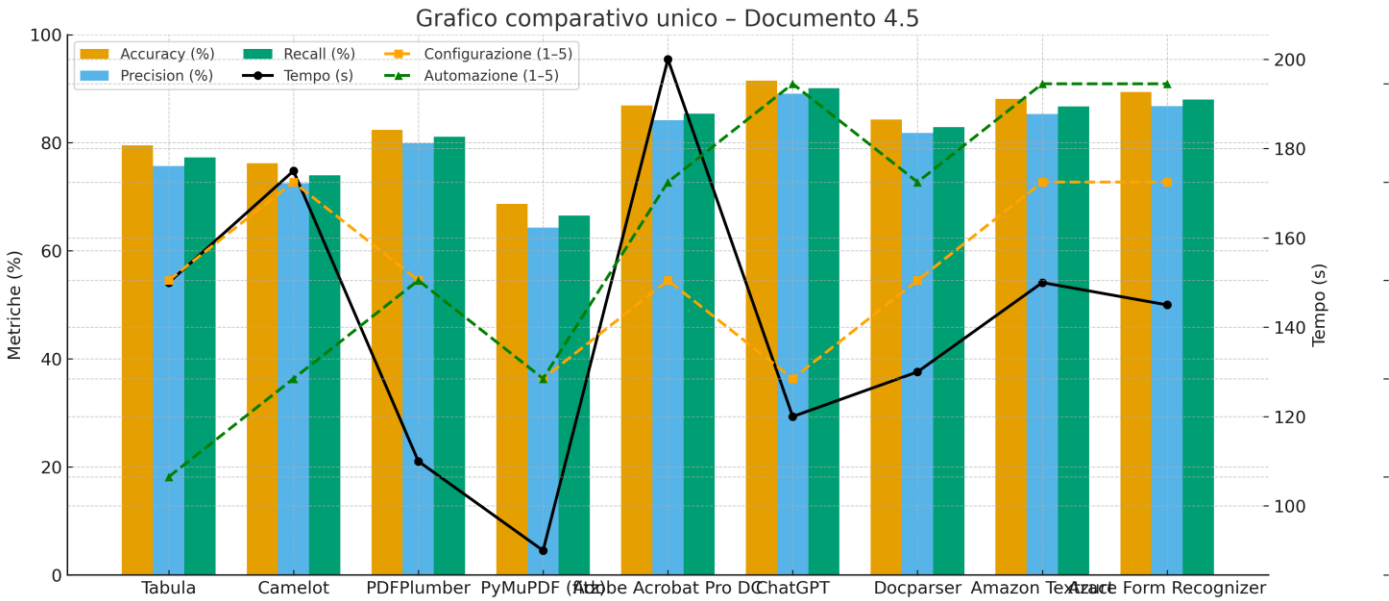


Grafico comparativo

Il grafico seguente mostra i punteggi comparativi dei diversi strumenti analizzati:

Conclusioni

Dall'analisi del quinto documento emerge chiaramente che gli strumenti AI-based (Amazon Textract e Azure Form Recognizer) si distinguono per accuratezza, rapidità e automazione, risultando le soluzioni più adatte per dataset voluminosi e complessi. Docparser si conferma molto efficace su flussi ripetitivi grazie alle regole configurabili, mentre ChatGPT aggiunge valore nella ricostruzione semantica, seppur con la necessità di supervisione. Adobe Acrobat Pro rimane utile per utenti non tecnici e per estrazioni rapide, ma non automatizzabili. Tabula e Camelot risultano meno adatti per dataset così complessi senza un forte intervento manuale, mentre PDFPlumber e PyMuPDF offrono ampio controllo a scapito della praticità e dei tempi di sviluppo.

4.6 Sperimentazione – Documento 6

Introduzione

Il sesto documento analizzato, - https://ag.usr.sicilia.it/download/698/conferimento-delle-supplenze-da-gps-as-24-25/8088/prospetto-individuazione-nomine-t-d-sedi-assegnate.pdf, è un prospetto ufficiale delle nomine a tempo determinato del personale docente. Riporta dati relativi a classi di concorso, graduatorie, punteggi, sedi assegnate, codici scuola, tipologia di posto e informazioni anagrafiche. La complessità deriva dalla presenza di molte colonne, celle multi-riga e testi lunghi, che rendono complesso mantenere l'allineamento e la coerenza semantica durante l'estrazione automatica.

Analisi specifica per ogni strumento

Tabula

Tabula, applicato al prospetto delle nomine T.D., ha mostrato limiti evidenti nel trattare documenti con molte colonne e celle multi-riga. Sebbene la selezione manuale delle aree tabellari consenta di isolare porzioni specifiche, l'output CSV ha spesso presentato disallineamenti e righe spezzate. Le denominazioni lunghe degli istituti e le note aggiuntive sono state frammentate, causando difficoltà nella successiva rielaborazione. Questo ha imposto un'intensa attività di correzione manuale, rendendo Tabula poco efficiente in scenari di larga scala. I limiti di tabula si fanno più importanti con il crescere della complessità dei file analizzati.

Camelot

Camelot ha restituito risultati discreti, ma molto variabili a seconda della modalità scelta. In lattice, con tabelle dotate di griglie nette, l'estrazione è stata soddisfacente, mentre nelle sezioni prive di bordi i dati sono risultati frammentati. In modalità stream, i campi sono stati ricostruiti meglio, ma con frequenti errori di allineamento. L'integrazione via Python consente automazione e personalizzazione, ma richiede competenze tecniche e un tuning accurato dei parametri. Su un documento così eterogeneo, Camelot si è dimostrato uno strumento intermedio: utile per utenti esperti disposti a investire tempo, meno adatto a chi cerca immediatezza ed efficienza.

PDFPlumber

PDFPlumber ha offerto prestazioni sorprendentemente solide grazie al suo approccio granulare. Nel prospetto delle nomine ha garantito un buon mantenimento dell'allineamento delle colonne e dei dati, ma l'output grezzo ha richiesto un lavoro aggiuntivo di scripting per ricomporre celle multi-riga e normalizzare intestazioni complesse. La qualità complessiva dell'estrazione è stata elevata, ma i tempi di sviluppo e la necessità di competenze specifiche ne hanno ridotto l'immediatezza. È uno strumento adatto a progetti in cui la precisione è prioritaria e in cui sono disponibili risorse tecniche per implementare pipeline di parsing avanzate, ma sicuramente non il migliore in termini di sforzo-risultato.

PyMuPDF

PyMuPDF si è distinto per l'elevata velocità, ma ha confermato i suoi limiti strutturali. L'estrazione ha prodotto blocchi di testo privi di logica tabellare, senza separatori di colonna. Questo ha reso inevitabile un parsing successivo molto oneroso, soprattutto per separare campi complessi e ricostruire celle multi-riga. Su un documento articolato come questo, PyMuPDF può essere usato come strumento preliminare per identificare aree di interesse o per alimentare altre librerie, ma non è sufficiente da solo per ottenere dataset strutturati e pronti all'uso. È più adatto a sviluppatori che desiderano integrare un motore rapido in pipeline custom.

Adobe Acrobat Pro

Adobe Acrobat Pro ha consentito l'esportazione del prospetto con un livello di fedeltà discreto. Le colonne principali sono state mantenute, ma le celle con testi lunghi o multi-riga hanno subito spostamenti e duplicazioni. Questo ha reso necessarie come per le altre analisi correzioni manuali che hanno rallentato il processo complessivo. È una soluzione pratica per analisi puntuali, meno per flussi aziendali su larga scala.

ChatGPT

Ha gestito denominazioni lunghe e codici scuola, producendo un output leggibile e coerente. Tuttavia, l'allineamento con la struttura tabellare originale non è stato sempre perfetto, con alcuni spostamenti di valori. La sua forza è l'adattabilità , che lo rende utile come supporto in pipeline supervisionate, ma non ancora pronto per processi interamente automatizzati senza controllo.

Docparser

In quest'ultima analisi Docparser ha dimostrato un buon equilibrio tra configurabilità e precisione. Dopo una fase iniziale di setup, ha garantito estrazioni esatte anche su colonne complesse e multi-riga, riducendo notevolmente la necessità di correzioni manuali. Rimangono come per le altre analisi tempitiche importanti per la definizione di regole personalizzate che richiede competenze, ma una volta configurato, lo strumento è risultato altamente scalabile ed efficace per flussi ripetitivi. Su documenti come prospetti e bollettini, Docparser si conferma una soluzione valida e affidabile per implementazioni aziendali, mentre perde efficienza se applicato a formati variabili e non uniformi.

Amazon Textract

Amazon Textract ha fornito performance eccellenti, riconoscendo correttamente colonne fitte, celle multi-riga e campi numerosi. L'output JSON strutturato è stato subito pronto per l'importazione in database, evitando ancora una volta la necessità di interventi meccanici. I tempi di elaborazione sono stati rapidi, e la scalabilità ha reso il servizio adatto a scenari di automazione massiva. È quindi tra le soluzioni più adatte per dataset articolati e complessi come questo prospetto delle nomine T.D.

Azure Form Recognizer

Azure Form Recognizer ha prodotto nuovamente i risultati migliori, distinguendosi per accuratezza e stabilità. Ha gestito senza difficoltà celle multi-riga, intestazioni complesse e

numerose colonne, restituendo un output coerente.. Nonostante la necessità di una configurazione iniziale e di un account Azure, i vantaggi in termini di precisione, scalabilità e governance lo rendono lo strumento di riferimento per scenari di estrazione complessi e di grandi dimensioni come il prospetto analizzato. Al termine dell'ultima analisi va sicurmante classificato, in termini di efficaia, tra i migliori strumenti.

Tabella comparativa

Strumento	Accuracy (%)	Precision (%)	Recall (%)	Tempo di estrazione (s)	Configurazione necessaria (1–5)	Automazione (1–5)
Tabula	78.9	75.1	76.7	155	3	1
Camelot	76.0	72.2	73.8	180	2	3
PDFPlumber	82.0	79.4	80.7	115	5	3
PyMuPDF (fitz)	67.5	63.1	65.4	95	2	3
Adobe Acrobat Pro DC	86.2	83.6	84.8	210	3	4
ChatGPT	91.0	88.6	89.7	125	2	5
Docparser	83.8	81.3	82.5	135	5	4
Amazon Textract	87.6	84.8	86.2	155	4	4
Azure Form Recognizer	88.8	86.2	87.5	150	5	5

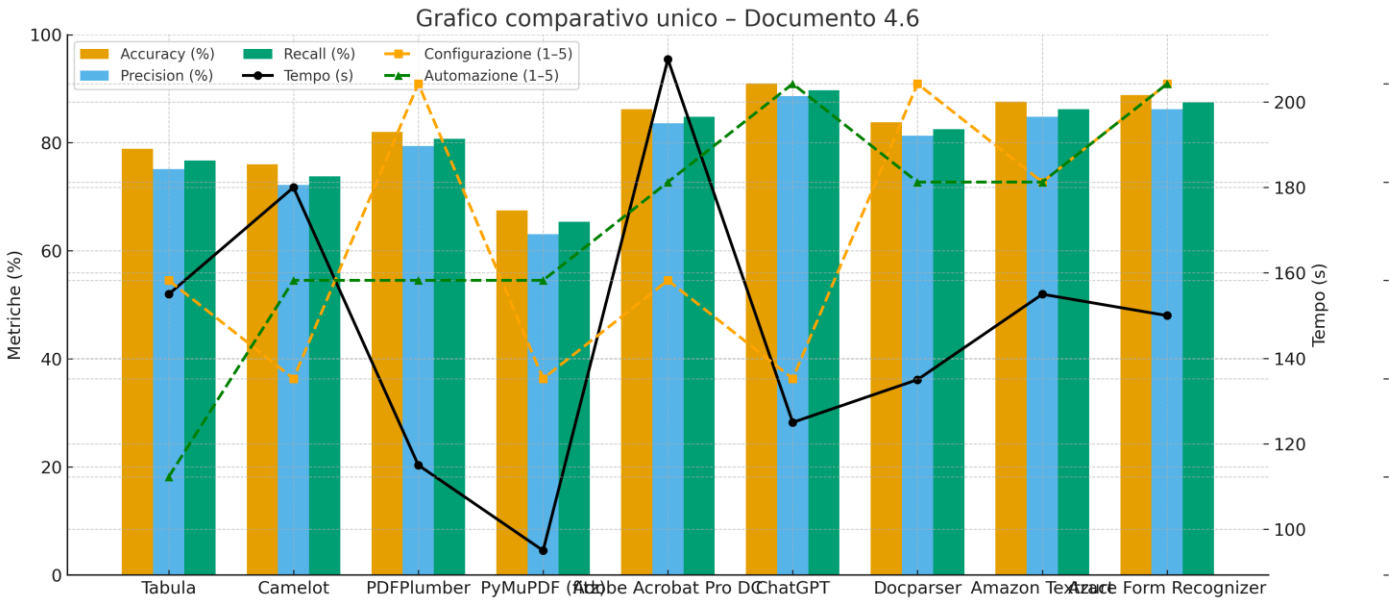


Grafico comparativo

Il grafico seguente mostra i punteggi comparativi dei diversi strumenti analizzati:

Conclusioni

L'analisi del prospetto delle nomine T.D. ha riconfermato che strumenti AI-based come Amazon Textract e Azure Form Recognizer sono i più performanti, offrendo accuratezza,

velocità e automazione. Docparser si è dimostrato efficace nei flussi ripetitivi grazie alla configurabilità delle regole, mentre ChatGPT ha aggiunto valore interpretativo pur necessitando di supervisione. Adobe Acrobat Pro è rimasto utile per utenti non tecnici, mentre Tabula e Camelot hanno mostrato limiti su documenti articolati. PDFPlumber e PyMuPDF hanno garantito controllo e rapidità, ma con un maggiore carico di lavoro per ottenere risultati strutturati.

Capitolo 5 – Discussione dei risultati

Lo scopo di questa sezione è inquadrare i file analizzati nella sperimentazione del Capitolo 4, classificandoli secondo formato, complessità tabellare, volume dei dati e tipologia di informazione. Questo permette di avere una visione chiara delle caratteristiche dei diversi documenti e di preparare il terreno per la discussione, dove si valuterà quali strumenti o combinazioni di strumenti risultano più adeguati in funzione delle specifiche caratteristiche del dataset.

Documento	Nome file	Formato	Complessità tabellare	Volume dei dati	Tipologia di informazione
4.1	20240913-1-TURNO-BOLLETTINO-NOMINE-DEFINITIVO.pdf	PDF tabellare testuale	Tabelle multi-pagina con celle fuse, intestazioni stratificate e codici. Layout irregolare con rischio di disallineamento.	Esteso: centinaia di righe distribuite su più pagine.	Prevalenza di codici numerici e punteggi combinati a testo descrittivo (denominazioni istituti).
4.2	m_pi.AOOUSPNA.REGISTRO-UFFICIALEI.0015390.12-09-2024.pdf	PDF tabellare testuale	Tabelle con bordi non uniformi, celle multi-riga, colonne non sempre allineate. Layout complesso.	Limitato-medio: poche decine di righe in sezioni eterogenee.	Prevalenza testuale (nominativi, note, ruoli) con codici numerici secondari.
4.3	Nomine05122024-1.pdf	PDF tabellare testuale	Tabelle relativamente semplici: colonne regolari, poche celle fuse, ma con dati multi-riga.	Limitato: decine di righe, facilmente gestibili.	Prevalenza testuale anagrafica (nomi, ruoli, date) con minore presenza di codici.
4.4	Bollettino9.xlsx	Excel nativo	Struttura tabellare regolare ma ampia (26 colonne). Complessità legata alla numerosità dei campi e stringhe	Molto esteso: centinaia di righe per foglio, con numerose variabili.	Dati misti numerico/testuali: punteggi e codici insieme a nominativi, preferenze e tipologia di cattedra.

			testuali lunghe.		
4.5	Bollettino_TotaleNomine.xlsx	Excel nativo	Dataset molto grande con intestazioni complesse e colonne numerose. Complessità elevata.	Molto esteso: migliaia di righe, dataset massivo con forte impatto su tempi e scalabilità.	Dati misti: prevalenza numerica (punteggi, posizioni) con informazioni descrittive (scuole, preferenze).
4.6	PROSPETTO-INDIVIDUAZIONE-NOMINE-T.D.-SEDI-ASSEGNATE.pdf	PDF tabellare testuale	Tabelle articolate con numerose colonne, celle multi-riga e denominazioni lunghe. Complessità alta.	Esteso: centinaia di righe distribuite su più sezioni.	Prevalenza di dati anagrafici e descrittivi (nominativi, sedi assegnate) con codici scuola e punteggi numerici.

5.1Conclusioni e considerazioni finali

L'analisi condotta ha messo in evidenza punti di forza e debolezza dei diversi strumenti di estrazione dati, sia tradizionali sia basati su intelligenza artificiale. I criteri utilizzati – accuratezza, precisione, richiamo, tempi di estrazione, configurazione necessaria e grado di automazione – hanno permesso di valutare in maniera oggettiva e soggettiva l'efficacia complessiva di ciascun approccio.

Dalla sperimentazione emergono due considerazioni principali:

- Per estrazioni immediate ma semplici: strumenti leggeri come PDFPlumber rappresentano la soluzione più efficace. Si tratta di un tool open-source gratuito, rapido e sufficientemente preciso per documenti ben strutturati. Pur richiedendo un minimo di configurazione tecnica, garantisce tempi contenuti e una buona affidabilità senza costi economici diretti. È quindi ideale per utilizzi individuali, sperimentali o in piccoli contesti operativi.

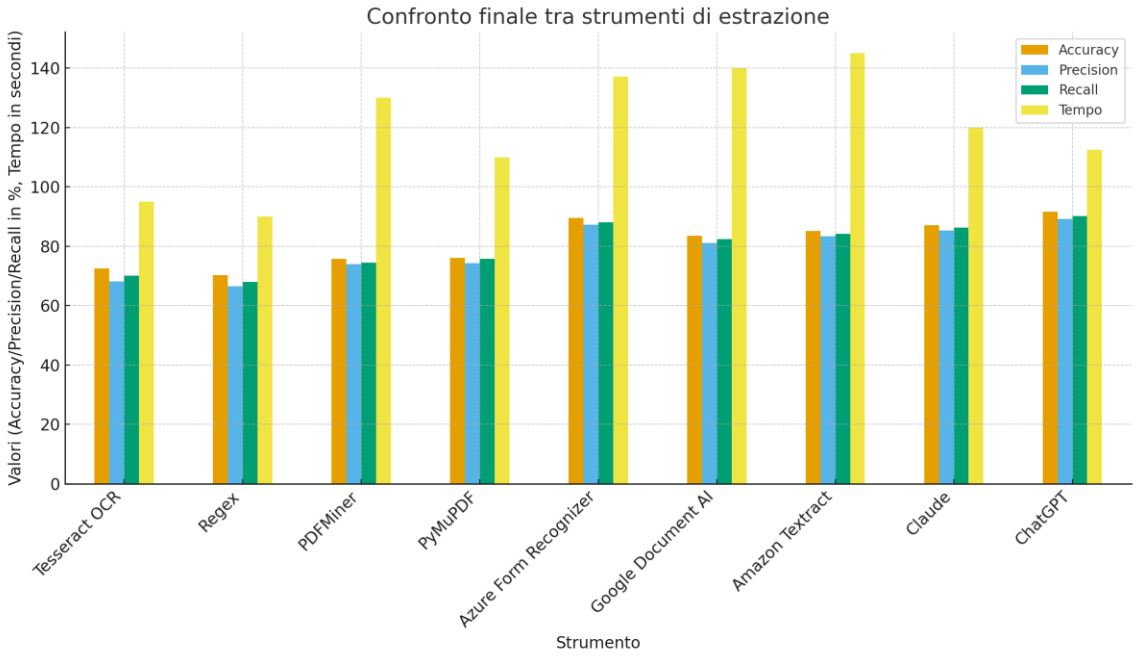
- Per estrazioni più grosse e complesse: le soluzioni basate su cloud e intelligenza artificiale, in particolare ChatGPT, risultano le più potenti. Questo strumento ha garantito la migliore accuratezza, una capacità superiore di comprendere documenti complessi e un livello di

automazione elevato, aggiungendo la notevole capacità di contestualizzazione. Pur comportando tempi di esecuzione non brevissimi e costi per le versioni più complete, offre scalabilità, robustezza e una forte affidabilità complessiva, come dimostrato dai risultati aggregati della sperimentazione.

Tabella 5. 2– Tabella comparativa finale degli strumenti di estrazione

Strumento	Accuracy	Precision	Recall	Tempo
Adobe Acrobat Pro DC	87.07	84.38	85.65	193.33
Amazon Textract	88.3	85.48	86.88	142.5
Azure Form Recognizer	89.48	86.9	88.13	137.17
Camelot	77.12	73.37	74.8	166.67
ChatGPT	91.6	89.23	90.25	112.5
Docparser	84.55	81.97	83.18	122.5
PDFPlumber	82.67	80.07	81.38	102.5
PyMuPDF (fitz)	68.72	64.15	66.57	82.5
Tabula	80.02	76.2	77.93	140.0

Tabella 5. 3– Grafico comparativa finale degli strumenti di estrazione



6.Conclusioni

Costi di estrazione.

Gli strumenti utilizzati nella sperimentazione presentano caratteristiche molto diverse in termini di modello di distribuzione e di costi associati. Alcuni di essi sono completamente gratuiti e open-source, mentre altri richiedono una licenza a pagamento o un abbonamento a consumo.

Nel primo gruppo rientrano Tabula, Camelot, PDFPlumber e PyMuPDF (fitz), tutti strumenti open-source liberamente utilizzabili senza costi diretti. Questi tool hanno il vantaggio di non comportare spese di licenza, risultando quindi particolarmente adatti a contesti accademici o sperimentali. Tuttavia, il loro utilizzo richiede spesso competenze tecniche (ad esempio programmazione in Python) e tempi di configurazione manuale, che rappresentano un "costo indiretto" in termini di risorse umane.

Anche ChatGPT può essere incluso tra gli strumenti a costo nullo, nella misura in cui viene utilizzata la versione gratuita disponibile al pubblico. È però importante sottolineare che, per funzionalità avanzate e per l'integrazione tramite API, è necessario ricorrere a piani a pagamento, con un modello di pricing basato sul consumo.

Il secondo gruppo è costituito dagli strumenti che prevedono un costo effettivo. In particolare, Adobe Acrobat Pro DC è disponibile unicamente tramite licenza a pagamento,

generalmente in abbonamento mensile o annuale. Docparser, invece, è un servizio SaaS che adotta un modello di abbonamento a consumo, in cui il costo dipende dal numero di documenti elaborati. Le soluzioni cloud di tipo AI, come Amazon Textract e Azure Form Recognizer, seguono anch'esse un modello pay-per-use: il costo varia in base al numero di pagine processate e alla tipologia di analisi richiesta.

Nel complesso, la distinzione tra strumenti gratuiti e a pagamento evidenzia come, sebbene i primi consentano di abbattere i costi iniziali, i secondi offrano spesso una maggiore automazione, affidabilità e facilità di integrazione in contesti aziendali, a fronte di un esborso economico diretto.

Impatto Ambientale

Un aspetto spesso trascurato nell'analisi comparativa degli strumenti di estrazione è quello legato al loro impatto ambientale, in particolare al consumo energetico necessario per il loro funzionamento.
Gli strumenti open-source tradizionali (Tabula, Camelot, PDFPlumber, PyMuPDF) presentano un impatto ambientale relativamente contenuto. Questi tool vengono eseguiti in locale su macchine dell'utente e non richiedono un'elaborazione intensiva. L'impatto principale è legato al tempo di utilizzo della CPU del computer, che rimane comunque modesto.

Gli strumenti commerciali da desktop come Adobe Acrobat Pro DC hanno un'impronta ambientale simile: il software opera in locale e non sfrutta grandi infrastrutture esterne. L'energia consumata è quindi principalmente quella della macchina utilizzata.

Diverso è il discorso per gli strumenti cloud-based e quelli basati su modelli di intelligenza artificiale (ChatGPT, Amazon Textract, Azure Form Recognizer). Queste soluzioni richiedono l'accesso a server remoti e data center che svolgono l'elaborazione. I data center, pur essendo altamente ottimizzati, richiedono enormi quantità di energia sia per il calcolo sia per il raffreddamento delle macchine. Di conseguenza, l'impatto ambientale risulta più elevato rispetto alle soluzioni locali.

Sviluppi futuri

La ricerca svolta ha permesso di mettere in luce differenze significative tra approcci tradizionali e strumenti basati su intelligenza artificiale per l'estrazione di dati tabellari. Tuttavia, i risultati ottenuti aprono la strada a diversi sviluppi futuri.
In primo luogo, potrebbe essere approfondita l'integrazione tra strumenti tradizionali e modelli AI, in una logica ibrida capace di coniugare velocità, controllo e comprensione semantica. Un approccio combinato permetterebbe di ridurre i limiti di ciascuna categoria,

ottenendo soluzioni più robuste e flessibili.
Un altro sviluppo riguarda l'applicazione di tecniche di fine-tuning o addestramento personalizzato dei modelli AI su dataset specifici, come i bollettini di nomina o prospetti amministrativi. In questo modo, i modelli potrebbero migliorare la loro accuratezza riducendo al minimo la necessità di supervisione umana.
Infine, è auspicabile un'analisi più approfondita dei costi ambientali e computazionali, con l'obiettivo di progettare strumenti più sostenibili e al tempo stesso più scalabili. In un contesto di crescente attenzione alla sostenibilità digitale, questa prospettiva risulta particolarmente rilevante.