SCOUTBOT

Interacting with structured and unstructured scouting reports.

Agenda

01. Data source

02. Data Preprocessing, Chunking and Embedding Strategy

03. Retreival Augmented Q/A Pipeline

04. Unstructured Data

05. Dealing with Unstructured Data

06. Future Scope

Data Source

- Data for this project was downloaded from the GitHub page of “Trouble with the Curve”, which is a CMU research paper focusing on ~10000 scouting reports till 2019.
- Data overview :

#	Column	Non-Null Count	Dtype
0	name	9175 non-null	object
1	key_mlbam	9175 non-null	object
2	key_fangraphs	9175 non-null	object
3	age	9175 non-null	float64
4	year	9175 non-null	int64
5	primary_position	9175 non-null	object
6	eta	9175 non-null	int64
7	report	9175 non-null	object
8	Arm	9175 non-null	int64
9	Changeup	9175 non-null	int64
10	Control	9175 non-null	int64
11	Curveball	9175 non-null	int64
12	Cutter	9175 non-null	int64
13	Fastball	9175 non-null	int64
14	Field	9175 non-null	int64
15	Hit	9175 non-null	int64
16	Power	9175 non-null	int64
17	Run	9175 non-null	int64
18	Slider	9175 non-null	int64
19	Splitter	9175 non-null	int64
20	source	9175 non-null	object
21	birthdate	9175 non-null	object
22	mlb_played_first	9175 non-null	int64
23	debut_age	9175 non-null	float64
24	label	9175 non-null	int64
25	text	9175 non-null	object

- Size: 9,175 scouting reports covering amateur draft, international, & MiLB prospects.
- Scope: Seasons 2013-2019 (year).
- Two Information Layers: “report” - scout report and “text” - fallback article.
- Pitcher-related columns: “Fastball”,” Slider”, “Changeup”, “Control”, etc.
- Hitter-related columns: “Hit”, “Power, Run”, “Field”, “Arm”.
- Player Metadata - “Name”, “primary_position”

Data Preprocessing, Chunking and Embedding Strategy

- 1 Row ➜ Rich Text Blob – merge narrative report with inline tool grades (e.g., “Fastball 60 Slider 55”) so numeric traits become searchable by the embedding model.
- Smart Split Rule – only rows > 1 600 chars get chunked; RecursiveCharacter splitter makes ~400-token segments with 50-token overlap to preserve sentence context.
- High-Recall Embeddings – OpenAI text-embedding-3-large (3 072-dim) encodes each chunk; numbers + prose live in the same vector space.
- Metadata Attached, Not Embedded – name, year, pos, eta, source stored as Pinecone metadata for fast filter queries (e.g., pos = "LHP", year = 2018).
- Pinecone Vector Store – ~9 k rows → ~9 k chunks → upserted to a 3 072-dim cosine index; query retrieves k nearest chunks, then GPT-4o answers with strictly grounded context.

Retreival Augmented Q/A Pipeline

- **User Question:** Free-form query ("Which left-handed 2018 arms show plus command + CH ⩾ 50?")
- **Metadata Filter** (optional): pos = "LHP", year = 2018 applied inside Pinecone → trims search space.
- **Vector Search** (Pinecone, k = 4): Cosine similarity on 3 072-dim embeddings → returns top-k chunk IDs + scores.
- **Context Builder** : Concatenate retrieved chunks with --- dividers → ~2-4 × 400 tokens
- **System prompt enforces**: "Use context only, obey numeric cut-offs, no hallucination."
- **GPT-4o Answer**: LLM reasons over context → returns concise list; code then prints the ranked source Preview

Typical latency: ~2.3 s end-to-end • Cost: ≈ $0.0006 per query
Grounding gap closed: answer always traceable to cited scouting-report snippets

Unstructured Data for RAG

Logo

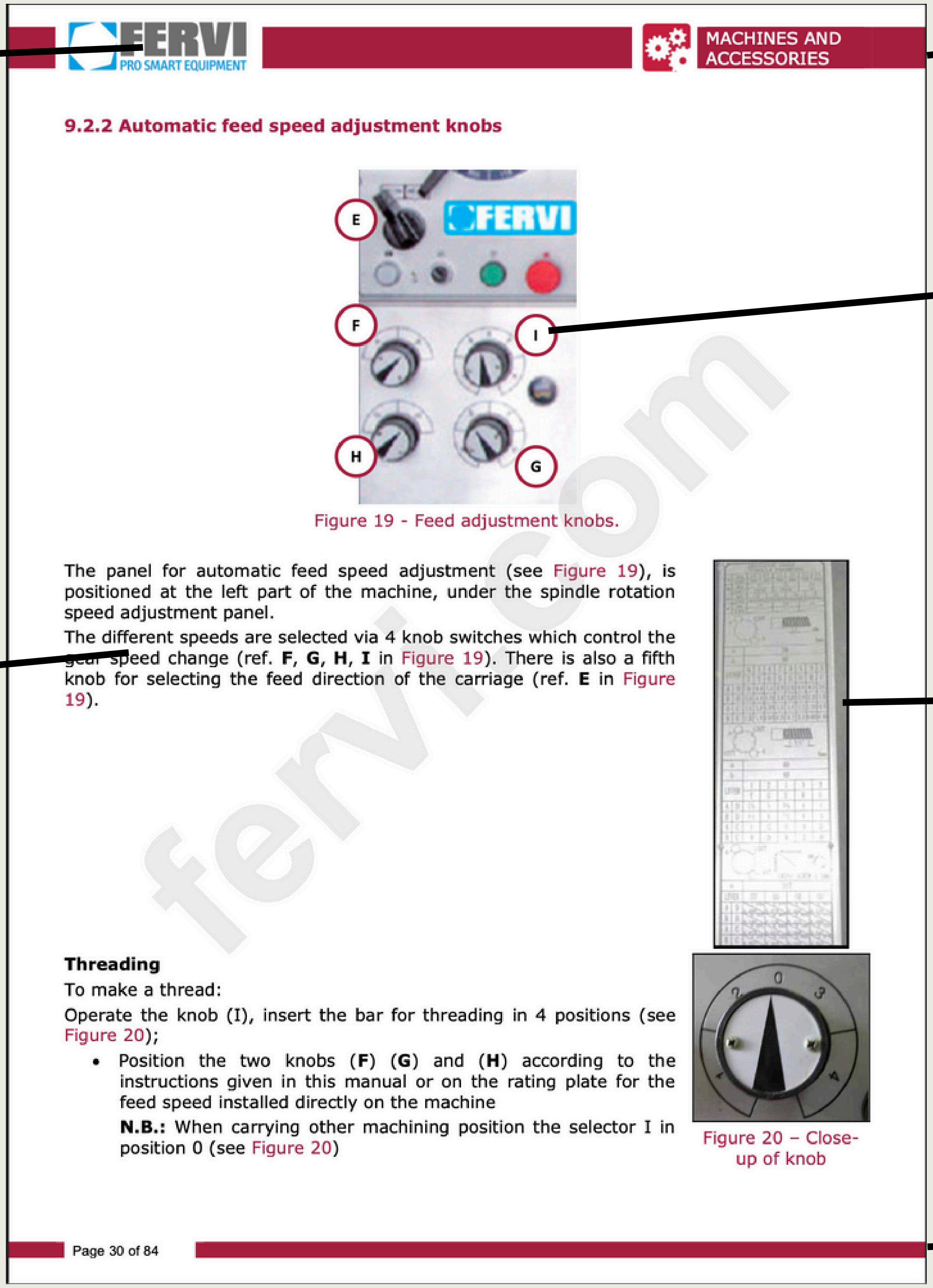
Header

Other objects

Text

Images

Footer



FERVI PRO SMART EQUIPMENT | MACHINES AND ACCESSORIES

9.2.2 Automatic feed speed adjustment knobs

Figure 19 - Feed adjustment knobs.

The panel for automatic feed speed adjustment (see Figure 19), is positioned at the left part of the machine, under the spindle rotation speed adjustment panel.
The different speeds are selected via 4 knob switches which control the gear speed change (ref. **F**, **G**, **H**, **I** in Figure 19). There is also a fifth knob for selecting the feed direction of the carriage (ref. **E** in Figure 19).

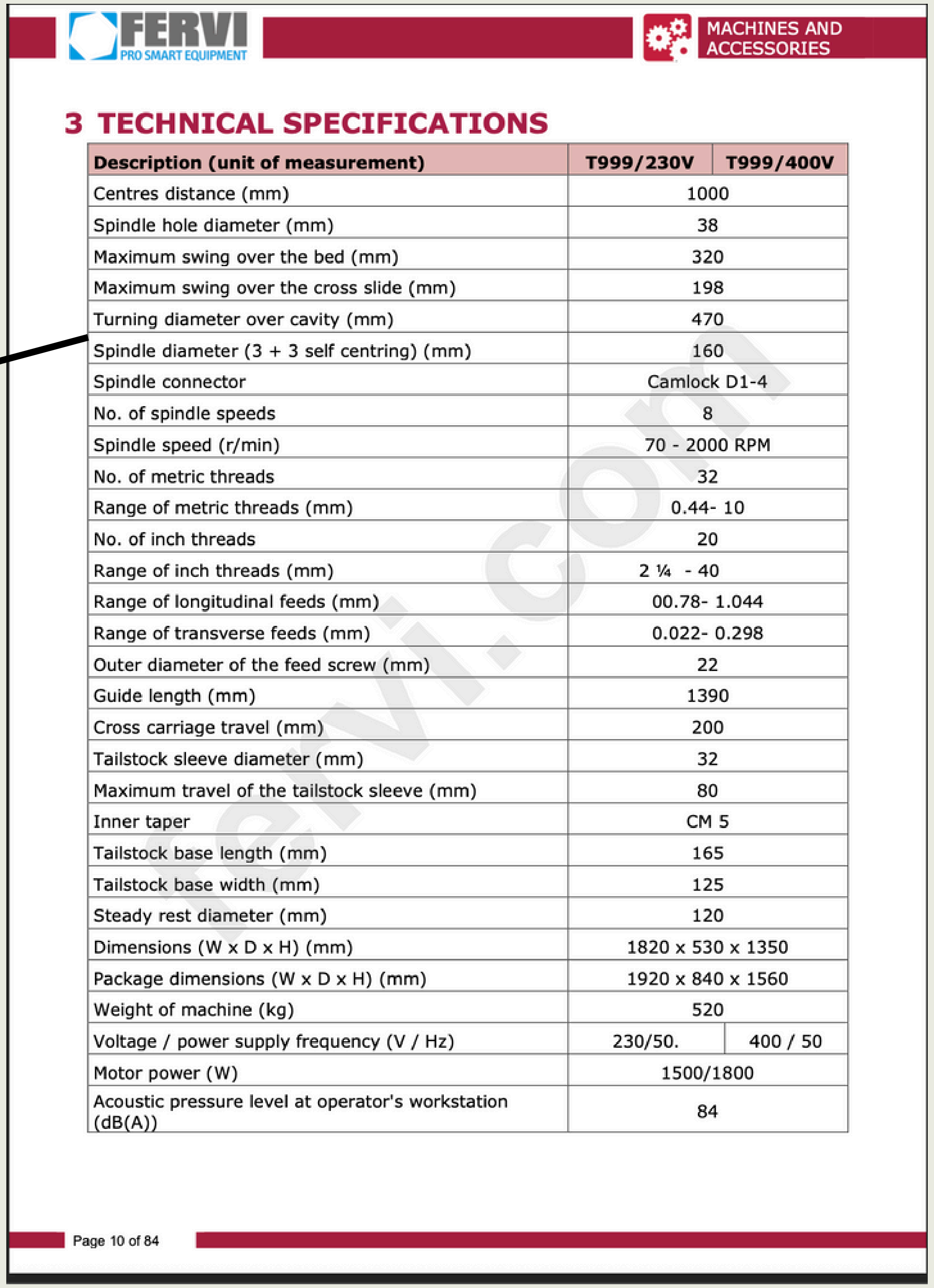
Threading
To make a thread:
Operate the knob (I), insert the bar for threading in 4 positions (see Figure 20);

- Position the two knobs (**F**) (**G**) and (**H**) according to the instructions given in this manual or on the rating plate for the feed speed installed directly on the machine
  N.B.: When carrying other machining position the selector I in position 0 (see Figure 20)

Figure 20 – Close-up of knob

Page 30 of 84

Tables



FERVI PRO SMART EQUIPMENT | MACHINES AND ACCESSORIES

3 TECHNICAL SPECIFICATIONS

Description (unit of measurement)	T999/230V	T999/400V
Centres distance (mm)	1000	
Spindle hole diameter (mm)	38	
Maximum swing over the bed (mm)	320	
Maximum swing over the cross slide (mm)	198	
Turning diameter over cavity (mm)	470	
Spindle diameter (3 + 3 self centring) (mm)	160	
Spindle connector	Camlock D1-4	
No. of spindle speeds	8	
Spindle speed (r/min)	70 - 2000 RPM	
No. of metric threads	32	
Range of metric threads (mm)	0.44- 10	
No. of inch threads	20	
Range of inch threads (mm)	2 ¼ - 40	
Range of longitudinal feeds (mm)	00.78- 1.044	
Range of transverse feeds (mm)	0.022- 0.298	
Outer diameter of the feed screw (mm)	22	
Guide length (mm)	1390	
Cross carriage travel (mm)	200	
Tailstock sleeve diameter (mm)	32	
Maximum travel of the tailstock sleeve (mm)	80	
Inner taper	CM 5	
Tailstock base length (mm)	165	
Tailstock base width (mm)	125	
Steady rest diameter (mm)	120	
Dimensions (W x D x H) (mm)	1820 x 530 x 1350	
Package dimensions (W x D x H) (mm)	1920 x 840 x 1560	
Weight of machine (kg)	520	
Voltage / power supply frequency (V / Hz)	230/50.	400 / 50
Motor power (W)	1500/1800	
Acoustic pressure level at operator's workstation (dB(A))	84	

Page 10 of 84

Solution : Multimodal RAG!

Dealing with Unstructured Scouting Reports

- **Unified Loader**: *PyPDFLoader* pulls every page → one Document per page with page # metadata.
- **Text Handling** - pdfminer-based extraction keeps paragraph order and heading levels. RecursiveCharacterTextSplitter respects “##/###” headers → ~400-token, 50-overlap chunks so sentences stay whole.
- **Images & Diagrams**: Non-text objects skipped during parse (no OCR needed for scouting PDFs). Figure captions are kept as plain text → preserved scouting notes on arm slots or biomechanics photos.
- **Tables**: In-line grade tables (e.g., “Fastball | 60”) flatten to ASCII rows; regex converts to “Fastball 60” strings so embeddings capture numeric meaning. Column borders & shading stripped to avoid token waste.
- **Noise Cleanup**: Footers like “Page 12 of 25” and watermark text removed via regex, set configuration for logos (example: DIAGRAM_THRESHOLD = 0.60, LOGO_THRESHOLD = 0.10)
- **Unicode tidy**: curly quotes → straight, em-dashes → “–”, ligatures split.
- **Rich Metadata Tags**: {"source_type": "pdf", "file": "2024-draft-catchers.pdf", "page":17} attached to every chunk for precise citation & filters.

Future Scope

- **All-in-one Lakehouse stack –** store our scouting-report vectors in Mosaic AI Vector Search, orchestrate the workflow in a Databricks notebook or job, and call GPT-4o through a model serving endpoint.

- **Agent Framework + LangChain** – Databricks' new Agent Framework lets us wrap each step (retrieve → reason → write) as tool-calling agents; LangChain integrations come pre-wired.

- **Model Context Protocol (MCP)** – Databricks hosts an MCP server that auto-registers our vector search and SQL helper functions as tools; any compliant agent can discover and invoke them without custom glue code.