**TextLab — User Manual**

A practical guide for non-technical users to download, extract, consolidate, analyze, and summarize evaluation reports with TextLab.

---

**1) What TextLab does**

TextLab helps you turn batches of PDF reports into structured text, then analyze them:

• Bulk download PDFs from links (CSV/Excel) — app.py

• Convert PDFs → TXT and consolidate by headings (user-provided) — app.py

• Keyword counting & semantic search over any text column — Word.py

• Topic modeling & semantic analysis — Document.py

• Summaries (overall, findings, keyword-focused) and a private Q&A chat — Summary.py

Everything runs locally. No external data is sent anywhere by default.

---

**2) Installation (once)**

1. Install Python 3.11+ (3.13 works; some optional libs install faster on 3.11).

2. In a terminal, create/activate a virtual environment and install dependencies:

python -m venv .venv

source .venv/bin/activate # Windows: .venv\Scripts\activate

pip install -U pip wheel

pip install streamlit pandas numpy scikit-learn altair requests rapidfuzz

pip install sentence-transformers bertopic umap-learn hdbscan rank-bm25

pip install pymupdf pdfplumber PyPDF2 langdetect argostranslate

3. (Optional offline translation) Install Argos Translate language models (e.g., es→en).

4. Launch TextLab:

streamlit run app.py

Tip: If SBERT/BERTopic is slow to build, switch to NMF on the Document page.

---

**3) Quick Start**

A. Bulk PDF Downloader (app.py)

1. Upload your CSV/Excel containing links.

2. Select the links column by name.

3. Choose a download folder and click Download PDFs.

4. (Optional) Create a ZIP of all PDFs and download the log CSV.

B. Convert & Consolidate (app.py)

1. Upload PDFs and click Convert to TXT.

2. In Consolidate & Normalize, enter your section headings (e.g., Findings, Conclusions).

3. Map country/year/region from your source table columns.

4. Assign regions for Unknown rows if needed.

5. Download the consolidated CSV.

C. Keyword Analyzer (Word.py)

1. Pick a text column (Findings of the evaluation or full\_text).

2. Paste keywords (comma or newline separated).

3. Choose Exact or AI/Semantic (SBERT/RapidFuzz expands synonyms).

4. Review counts by document, year/region charts, and validation snippets.

5. Download detailed table and totals.

D. Document Topics (Document.py)

1. Pick Whole documents or Keyword-centered.

2. Choose BERTopic (SBERT) or NMF/LDA.

3. Adjust min\_df/max\_df (auto-clamped).

4. Explore topics, semantic quality, top docs, heatmap, and semantic search.

5. Download topics and per-item assignments.

E. Summaries & Chat (Summary.py)

1. Generate overall summaries, findings summaries, and keyword-focused summaries.

2. Use the private chat to query by country/region/year grounded only in your consolidated data.

---

**4) Common Tasks**

• My PDFs say no\_text\_found. They are likely scanned images (add OCR later).

• Few keyword hits but I know they exist: switch to AI/Semantic, lower threshold (~0.45), or analyze full\_text.

• BERTopic finds few topics: lower min\_topic\_size, or switch to NMF; increase corpus size.

• Language varies: turn on auto-translate to English (Argos) in Options.

---

**5) Troubleshooting**

• InvalidParameterError: max\_df 1.001 → fixed; app clamps min\_df/max\_df.

• ModuleNotFoundError → install required libs in the same virtualenv.

• Slow extraction → prefer PyMuPDF and process in smaller batches.

• Empty results → verify text column, headings, or OCR.

---

**6) UX Tips**

• Shift+Enter adds a newline in text boxes.

• Sidebar controls tune thresholds and model choices.

• Chat input stays docked at the bottom.

---

**7) Privacy**

• Local processing only, no data leaves your machine.

• Exports are local CSV/ZIP — handle per policy.

---

**8) Glossary**

• BM25: lexical relevance.

• SBERT: sentence embeddings.

• BERTopic: clustering + embeddings.

• min\_df/max\_df: vectorizer frequency cutoffs.