Finding the Structure of Documents

- **Document Structuring** is a key subtask of **Natural Language Generation (NLG)**.
- It focuses on organizing information into a logical sequence, including **deciding sentence order, grouping text into paragraphs, and structuring content flow**. It is closely related to **Content Determination**, which involves selecting the information to be included in the generated text.

Two critical components of **document structuring** are:

1. **Sentence Boundary Detection**
2. **Topic Boundary Detection**

1. Sentence Boundary Detection (SBD)

- **Sentence Boundary Detection (SBD)** is the process of **identifying the end of a sentence** in a given text. It is crucial in NLP applications such as **text summarization, machine translation, and speech-to-text processing**.

Challenges in Sentence Boundary Detection

- SBD is not as simple as detecting periods (.) because **abbreviations, numbers, and formatting variations** can cause confusion.

Example of Sentence Boundary Ambiguity

Case 1: Abbreviations

- **Incorrect detection:**
- Dr. John is an expert in NLP. He has worked at Google Inc. since 2015.
- A naive SBD system might **incorrectly split after "Dr." and "Inc."**, assuming they are sentence boundaries.
- **Correct detection:**
- Dr. John is an expert in NLP.
  He has worked at Google Inc. since 2015.
- To correctly handle such cases, **machine learning models** or **rule-based systems** (such as regular expressions) are used.

Sentence Boundary Detection

Case 2: Numerical Values and Dates

Incorrect detection:

- The temperature in New York was 23.5 degrees yesterday. It will be lower today.
- A simple rule-based system might **mistakenly treat "23.5" as a sentence break**.

Correct detection:

- The temperature in New York was 23.5 degrees yesterday.
  It will be lower today.

Techniques for Sentence Boundary Detection

- **Rule-Based Methods** – Use **regular expressions** to identify punctuation patterns.
- **Statistical Methods** – Use **Hidden Markov Models (HMMs)** to learn sentence-ending probabilities.
- **Machine Learning Methods** – Train classifiers like **Naïve Bayes, Decision Trees, or Deep Learning** to distinguish sentence boundaries.

2. Topic Boundary Detection (TBD)

- **Topic Boundary Detection (TBD)** identifies where one topic **ends** and another **begins** in a document. This is crucial for **document summarization, information retrieval, and text segmentation**.

Challenges in Topic Boundary Detection

- Detecting topic changes is difficult because **topics can shift gradually or abruptly**, depending on the writing style.

Example of Topic Boundary Changes

Case 1: News Article

- Consider a **news report** with the following paragraphs:
- The stock market opened higher today, with major indices gaining points. Experts attribute the rise to positive earnings reports.
- Meanwhile, in sports, the local football team secured a victory against their rivals, thrilling fans.
- A **Topic Boundary Detection** system should recognize that **"Stock Market" and "Sports"** are separate topics.

Case 2: Research Paper

- A research paper might have the following sections:
- **Introduction** – Defines the problem and motivation.
  Related Work – Discusses previous research.
  Methodology – Explains the approach used.
  Results and Discussion – Presents findings and insights.
- A **TBD system** must correctly **segment** these sections.

Methods

- Document structuring and sentence segmentation involve techniques to determine **sentence boundaries, topic boundaries, and overall structure** in a text. Various machine learning approaches are used to accomplish this task.

The key methods include:

1. Generative Sequence Classification Methods
2. Discriminative Local Classification Methods
3. Hybrid Approaches
4. Discriminative Sequence Classification Methods
5. Extensions for Global Modeling for Sentence Segmentation

1. Generative Sequence Classification Methods

- These methods use **probabilistic models** that learn the **joint probability** of words and their corresponding labels (sentence boundaries or topic changes). One of the most common generative models is the **Hidden Markov Model (HMM)**.

Example: Hidden Markov Model (HMM) for Sentence Boundary Detection

- Consider this text:
- "Dr. Smith is an expert in AI. He works at Google Inc. in California."
- A **naïve rule-based system** might incorrectly split after "Dr." or "Inc.". An **HMM-based model** assigns probability scores to whether a word **ends a sentence** or not.

How it works

- **States**: Sentence boundary (B), non-boundary (NB).
- **Observations**: Words, punctuation, capitalization.
- **Transition probabilities**: P(NB → B), P(B → B), etc.
- 🔹 **Correct output (after HMM analysis):**
  ✅ "Dr. Smith is an expert in AI. | He works at Google Inc. in California."
- **Pros & Cons**
- ✅ Simple and interpretable.
  ❌ Cannot capture deep semantic relationships.

2. Discriminative Local Classification Methods

- Unlike generative models, **discriminative models** learn **decision boundaries** to classify **each punctuation mark** as a **sentence boundary (B) or non-boundary (NB)**.

Example: SVM or Logistic Regression for Sentence Segmentation

- Consider the sentence:
- "New York is beautiful. The weather is great!"
- A **local classifier** takes **each punctuation mark (. or !)** and decides whether it marks a sentence boundary.
- **Features used**
- **Previous and next words**: "beautiful", "The".
- **Punctuation type**: . or !.
- **Capitalization of the next word** (The is capitalized → likely a new sentence).
- 🔹 **Correct output:**

  ✅ "New York is beautiful. | The weather is great!"

3. Hybrid Approaches

- Hybrid models **combine generative and discriminative methods** for better accuracy. A common **hybrid approach** is **Conditional Random Fields (CRF) + Neural Networks**.

Example: CRF for Email Segmentation

- Consider an **email structure**:
- "Dear John,
- I hope you're doing well.
- Best regards,
  Alice"

A **CRF-based model** considers features like:

- **Line breaks** (indicating new sections).
- **Greetings (Dear) and signatures (Best regards)**.
- **Word embeddings** to detect sentence importance.
- 🔹 **Correct output (segmented email):**
  ✅ "Dear John, | I hope you're doing well. | Best regards, Alice"
- **Pros & Cons**
- ✅ More accurate than pure rule-based methods.
  ❌ Computationally expensive.

4. Discriminative Sequence Classification Methods

- These methods classify **entire sequences** rather than individual words, allowing the model to **learn context better**.
- **Example: LSTM for Sentence Segmentation in Chat Messages**
- Consider a **text message conversation**:
- "hey how are you? i am fine thanks. what about you?"
- A simple rule-based approach might fail to split correctly. An **LSTM-based model** learns sentence structure based on **word embeddings** and **contextual dependencies**.
- 🔹 **Correct segmentation:**
  ✅ "Hey, how are you? | I am fine, thanks. | What about you?"
- **Why is LSTM better?**
- It **remembers previous words**, helping in cases like:
  ✅ "I saw Mr. Brown today. He looked happy."
  (Avoids breaking after "Mr.").
- **Pros & Cons**
- ✅ Handles **long-range dependencies** well.
  ❌ Needs large training datasets.

5. Extensions for Global Modeling for Sentence Segmentation

- These methods consider **long documents** and optimize for **paragraph and document structuring**.
- **Example: Hierarchical Attention Network (HAN) for News Article Structuring**
- Consider a **news article**:
- "Stock markets rose today due to positive earnings reports. Experts predict further growth.
- Meanwhile, in sports, the local football team won their championship game."
- A **Hierarchical Attention Network (HAN)**:
- First **analyzes words** within sentences.
- Then **analyzes sentences** to determine **topic boundaries**.
- 🔹 **Correct segmentation:**

  ✅ "Stock markets rose today due to positive earnings reports. Experts predict further growth." ✅

  "Meanwhile, in sports, the local football team won their championship game."`
- **Pros & Cons**
- ✅ Best for **long documents** and **paragraph segmentation**.

  ❌ Requires high computational power.

Complexity of the Approaches

- The complexity of different approaches varies based on **time, memory, training, prediction, and feature extraction**. Here’s a summary:
- **1.1. Discriminative vs. Generative Models**
- **Discriminative Approaches (e.g., CRFs, SVMs, Neural Networks)**
  - 🔴 **Higher training complexity** (requires multiple passes over data).
  - 🔴 **Slower inference** (feature extraction is costly).
  - 🟢 **Performs well with fewer training samples**.
  - 🟢 **Handles diverse feature sets (e.g., words, POS tags, punctuation)**.
- **Generative Approaches (e.g., HMMs, Naïve Bayes, HELMs)**
  - 🟢 **Handles large datasets efficiently** (e.g., decades of news transcripts).
  - 🟢 **Faster prediction (fewer features, simpler models)**.
  - 🔴 **Poor at handling unseen events (limited feature set)**.

Example:

- **HMM (Generative Model)** predicts **sentence boundaries** using word probabilities.
- **CRF (Discriminative Model)** uses word features + POS tags + punctuation but is **slower** due to feature extraction.

Local vs. Sequence-Based Approaches

- **Local Approaches (Rule-based, SVMs, Decision Trees)**
  - 🟢 **Faster** (only analyzes single sentences).
  - 🔴 **Less accurate** (misses dependencies between sentences).
- **Sequence-Based Approaches (HMMs, CRFs, LSTMs)**
  - 🔴 **Complex due to decoding** (evaluates multiple sequences).
  - 🟢 **More accurate** (captures dependencies across sentences).

Example:

- **Local Approach:** Classifies each sentence **independently** (faster, but ignores context).
- **Sequence Approach:** Uses **previous and next sentences** for better accuracy (slower).

Polynomial vs. Exponential Complexity

- **Dynamic programming** helps sequence-based models run in **polynomial time** instead of exponential.
- Complexity grows **exponentially** with:
  - Number of **boundary candidates**.
  - Number of **sentence boundary states**.

Example:

- **CRF training complexity**: Requires multiple **inference passes** on training data (expensive).
- **HMM training complexity**: Uses **simple probability calculations** (faster)

Performance of Approaches

- Performance evaluation depends on **accuracy, error rate, F1-score, and recall**.

Evaluation Metrics

- **Error Rate** = (Number of errors) ÷ (Total sentences).
- **F1-score** = 2 × (Precision × Recall) ÷ (Precision + Recall).
- **NIST Error Rate** = (Wrong labels) ÷ (Actual boundaries).

Example:

- A **rule-based system** for sentence segmentation in **speech** may have a **higher error rate** due to speech ambiguities.
- A **deep learning system** (LSTMs) may have a **lower F1-score** if trained on **limited data**

1. Error Rate

$$\text{Error Rate} = \frac{\text{Number of errors}}{\text{Total sentences}}$$

- **Number of Errors:** The count of mistakes made by the system, such as incorrect classifications, mislabeling, or missing information.
- **Total Sentences:** The total number of sentences in the dataset.
- **Explanation:** This metric gives a simple proportion of errors relative to the total number of sentences. A lower value indicates better performance.

2. F1-Score

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- **Precision:** Measures how many of the retrieved results were actually correct.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

  - **True Positives (TP):** Correctly identified items.
  - **False Positives (FP):** Incorrectly identified items (false alarms).
- **Recall:** Measures how many of the actual correct items were retrieved.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

  - **False Negatives (FN):** Missed correct items.
- **Explanation:** F1-score balances precision and recall, making it a good metric when both false positives and false negatives matter.

NIST Error Rate – Detailed Explanation

NIST stands for the **National Institute of Standards and Technology.** It is a U.S. government agency that develops measurement standards, including evaluation metrics for various technologies such as speech recognition, text processing, and machine learning.

3. NIST Error Rate

$$\text{NIST Error Rate} = \frac{\text{Wrong labels}}{\text{Actual boundaries}}$$

- **Wrong Labels:** The number of incorrect classifications or label assignments in the output.
- **Actual Boundaries:** The true segmentation points or classifications present in the dataset.
- **Explanation:** This metric, often used in speech and text processing, measures how often the system incorrectly labels or segments data. A lower value indicates better accuracy.

Performance Comparison in Text Segmentation

- **Mikheev's Rule-Based Model**: **Error rate = 1.41%**.
- **With Abbreviation List**: **Error rate = 0.45%**.
- **With POS-based Classifier**: **Error rate = 0.31%**.
- **Gillick's SVM-based Model**: **Error rate = 0.25%** (best performance).

Key Takeaway:

- **Supervised ML (SVMs, CRFs) outperforms rule-based methods**.
- **Sentence segmentation errors** affect **subsequent NLP tasks** (e.g., summarization).

Summary Table

Approach	**Training Complexity**	**Prediction Speed**	**Accuracy**	**Best Use Cases**
Rule-Based	Low	Fast	Moderate	Simple structures, legal documents
HMM (Generative)	Medium	Fast	Moderate	Speech segmentation
SVMs (Discriminative)	High	Slow	High	Text classification, sentence segmentation
CRFs (Sequence-based)	Very High	Slowest	Very High	Complex NLP tasks (NER, POS tagging)
Deep Learning (LSTMs, BERT)	Highest	Slowest	Best	Large-scale NLP (summarization, translation)