Document Structure and Segmentation

1 Document Classification and Segmentation

Document classification is a supervised learning problem in which a piece of text is represented as a feature vector $x \in X$ and assigned a label $y \in Y$. In its simplest form, it is a binary classification problem, where the model decides between two classes (for example, boundary vs. non-boundary). More generally, it can also be extended to multi-class classification for fine-grained distinctions. Given training data $\{(x_i, y_i)\}_{i=1}^{n}$, the objective is to learn a function that can correctly assign labels to unseen data. From a probabilistic perspective, this means learning a model that predicts the label y that maximizes the conditional probability $P(y \mid x)$.

2 Sentence and Topic Segmentation

Segmentation tasks are closely related to classification but operate at a structural level. In sentence segmentation, the goal is to determine where one sentence ends and the next begins. In topic segmentation, the task is to identify where one topic ends and another starts within a document.

A common approach is to treat each word position as a candidate boundary. For every candidate position, the system assigns a label such as B (boundary) or NB (non-boundary). In this way, segmentation can initially be viewed as a classification problem where each position is independently labeled.

However, independent classification can be problematic because natural language exhibits structural constraints. For example, abbreviations such as "Dr." or "U.S.A." contain periods but do not indicate sentence boundaries. In spoken language, such as broadcast news, two sentence boundaries rarely occur consecutively. If decisions are made independently using only

local features, the model cannot effectively enforce such global structural constraints.

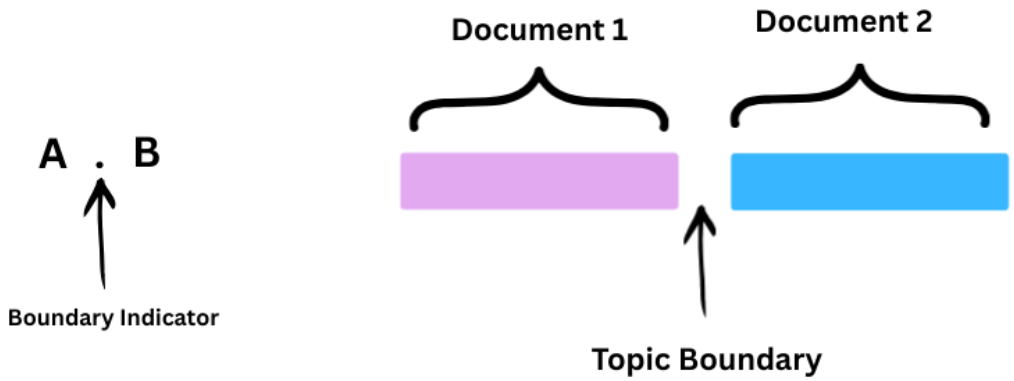

Figure 1: Boundary Indicator for Document and Topics segementation

3 Sequence Modeling for Segmentation

To address these limitations, segmentation is reformulated as a sequence classification problem. Instead of predicting each boundary independently, the model considers the entire sequence of inputs and labels jointly.

Let

$$X = (x_1, x_2, \ldots, x_n)$$

be the sequence of input features, and

$$Y = (y_1, y_2, \ldots, y_n)$$

be the corresponding sequence of labels. The objective is to find the most probable label sequence:

$$\hat{Y} = \arg\max_{Y} P(Y \mid X).$$

This formulation enables the model to capture dependencies between neighboring labels and to enforce constraints such as avoiding adjacent boundaries or ensuring that sentences contain at least a minimum number of words. Sequence models such as Hidden Markov Models (HMMs) and Conditional Random Fields (CRFs) are commonly used for such tasks.

4 Generative and Discriminative Models

There are two principal learning paradigms for classification and segmentation tasks: generative models and discriminative models.

In a generative model, the joint probability distribution

$$P(X, Y)$$

is estimated. From this joint distribution, the conditional probability $P(Y \mid X)$ is derived using Bayes' rule. Generative models require specific assumptions about how the data is generated and often use smoothing or back-off techniques to handle unseen events.

In contrast, a discriminative model directly estimates the conditional probability

$$P(Y \mid X).$$

These models focus on learning decision boundaries between classes and allow the use of rich, overlapping feature representations without making strong assumptions about the underlying data distribution. Discriminative models are particularly effective in segmentation tasks because they can better capture contextual dependencies between labels.

5 Hidden Markov Models (HMM)

A Hidden Markov Model (HMM) is based on the idea of a stochastic process in which the current state depends on previous states. In general, a stochastic process satisfies:

$$P(X_n \mid X_0, X_1, \ldots, X_{n-1}).$$

However, a Markov chain simplifies this dependency using the **Markov property**, which states that the current state depends only on the immediately previous state:

$$P(X_n \mid X_0, \ldots, X_{n-1}) = P(X_n \mid X_{n-1}).$$

Under this assumption, the joint probability of a sequence of states can be written as:

$$P(X_0, X_1, \ldots, X_n) = P(X_0) \prod_{i=1}^{n} P(X_i \mid X_{i-1}).$$

In a Markov chain, we define transition probabilities between states as in Figure 2 . For example, consider two states A and B with transition matrix:

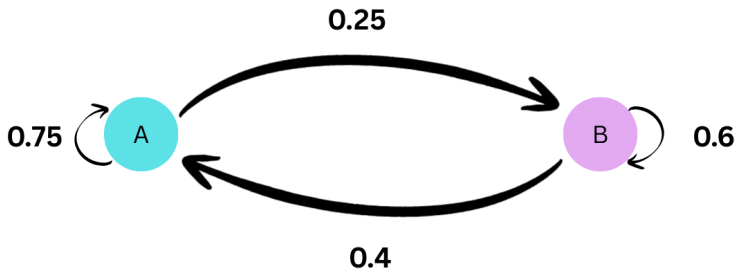

Figure 2: Transition Graph

$$P = \begin{bmatrix} 0.75 & 0.25 \\ 0.40 & 0.60 \end{bmatrix}.$$

If the initial state distribution is

$$S_0 = [1 \;\; 0],$$

then the state distribution after one step is:

$$S_1 = S_0 P = [0.75 \;\; 0.25].$$

After two steps:

$$S_2 = S_1 P = S_0 P^2.$$

In general, after n steps:

$$S_n = S_0 P^n.$$

An HMM extends a Markov chain by introducing **hidden states** and **observations**. The hidden states evolve according to transition probabilities, and each state emits observable outputs according to emission probabilities.

6 Generative Sequence Modeling with HMM

In generative sequence modeling, the goal is to find the most probable label sequence:

$$\hat{Y} = \arg\max_{Y} P(Y \mid X).$$

Using Bayes' rule:

$$\hat{Y} = \arg\max_{Y} \frac{P(X \mid Y)P(Y)}{P(X)}.$$

Since $P(X)$ is constant for all Y, we simplify:

$$\hat{Y} = \arg\max_{Y} P(X \mid Y)P(Y).$$

The likelihood term is decomposed as:

$$P(X \mid Y) = \prod_{i=1}^{n} P(x_i \mid y_1, \ldots, y_i),$$

and the prior over states as:

$$P(Y) = \prod_{i=1}^{n} P(y_i \mid y_1, \ldots, y_{i-1}).$$

In an HMM, we make two simplifying assumptions:

$$P(x_i \mid y_1, \ldots, y_i) \approx P(x_i \mid y_i),$$

$$P(y_i \mid y_1, \ldots, y_{i-1}) \approx P(y_i \mid y_{i-1}).$$

Thus, the model becomes a first-order Markov model with emissions depending only on the current state.

7 Bigram HMM for Segmentation

In the bigram case, the system is modeled as a fully connected m-state Markov model, where m is the number of boundary categories (e.g., B for boundary and NB for non-boundary). The hidden states represent boundary types, and each state emits words (or sentences, in topic segmentation).

The most likely hidden state sequence that generates the observed word sequence is computed using dynamic programming, typically the **Viterbi algorithm**.

For example, consider a bigram model with two states: B and NB. Given an observed word sequence such as:

people are dead few pictures,

the model estimates the most probable state sequence:

$$NB \;\; NB \;\; B \;\; NB \;\; NB.$$

8 Topic Segmentation with HMM

For topic segmentation, multiple states are used, where each state corresponds to a topic. If there are n topics, then n hidden states are defined. The challenge lies in estimating the observation likelihood:

$$P(X_i \mid Y_i),$$

especially when the topic categories are not explicitly known.

One approach is to use a unigram language model to estimate word probabilities within a topic. Another approach is to apply clustering methods such as k-means to group similar text segments and estimate state-dependent word distributions. These probabilities are then integrated into the HMM framework.

9 Limitations of HMMs

Although HMMs are powerful generative models, they have several limitations. They rely heavily on word sequences and make strong independence assumptions. They cannot easily incorporate additional linguistic features such as part-of-speech (POS) tags, prosodic cues (pause, pitch, duration) in speech, or other non-lexical information.

These limitations restrict their effectiveness in tasks such as speech recognition, segmentation, and richer linguistic analysis. Therefore, extensions or alternative models are often required to overcome these constraints.

10 Extensions to HMM for Boundary and Event Modeling

Shriberg suggested an extension to standard HMM-based segmentation models in which boundaries are treated as explicit states that emit boundary tokens. Instead of treating boundaries as implicit transitions between words, they become observable events generated by special hidden states. This modification allows the model to explicitly represent structural events such as sentence boundaries, repairs, and filled pauses. As a result, it becomes possible to integrate non-lexical cues and combine the model with other linguistic components.

11 Hidden Event Language Model (HELM)

The Hidden Event Language Model (HELM) was originally designed for modeling speech disfluencies, such as "uh", "um", and self-repairs. In this framework, disfluencies and boundary events are treated as meta-tokens inserted into the word sequence. These hidden events are generated by special states in the model.

The boundary model structure typically includes two special states:

- SB : Sentence (or segment) boundary
- NB : No boundary

The remaining states generate normal lexical words.

Imaginary Token Trick

A practical modeling trick is to insert an imaginary token between every pair of consecutive words. At each inserted position, the model decides whether a boundary event occurs. Formally, for a word sequence:

$$\text{I like NLP it is interesting}$$

we transform it into:

$$\text{I } [\circ] \text{ like } [\circ] \text{ NLP } [\circ] \text{ it } [\circ] \text{ is } [\circ] \text{ interesting}$$

At each $[\circ]$, the model predicts:

$$SB \quad \text{(boundary)} \quad \text{or} \quad NB \quad \text{(no boundary)}.$$

Thus, boundaries become explicit observable events rather than implicit structural assumptions.

12 Modeling Repairs and Filled Pauses

In spontaneous speech, repairs occur when a speaker starts saying something, interrupts it, and then corrects or restarts the utterance. For example:

I want to go to Del- uh- Mumbai tomorrow

This can be represented as:

I want to go to Del $\langle RP \rangle$ Mumbai tomorrow

where $\langle RP \rangle$ denotes a repair event.

Other possible event boundaries include:

- FP : Filled pause
- RP : Repair
- SB : Sentence boundary
- NB : No boundary

Repairs are not normal lexical words, but they strongly affect structure and meaning. Therefore, modeling them explicitly improves segmentation and speech understanding.

13 Factored Hidden Event Language Model (fHELM)

A further extension is the factored Hidden Event Language Model (fHELM). This model extends HELM by factoring complex hidden events into multiple

interpretable components. Instead of representing a repair event as a single atomic symbol, it decomposes it into attributes.

In standard HELM, we may have hidden tokens such as:

$$\langle RP \rangle, \quad \langle SB \rangle.$$

However, real speech has richer internal structure. For example, a repair may involve:

- Type of repair
- Position where it occurs
- What triggered it (pause, pitch change, word fragment)
- Scope of correction

In fHELM, a repair event can be represented as:

$$\langle \text{event-type} = \text{REPAIR}, \ \text{repair-type} = \text{WORD}, \ \text{interruption} = \text{TRUE}, \ \text{pause-length} = \text{SHORT} \rangle.$$

For example:

Book a flight on Fri- Saturday

can be modeled as:

Book a flight on Fri $\langle RP \rangle$ Saturday

and in the factored representation:

$$\langle \text{type} = \text{REPAIR}, \ \text{cue} = \text{FILLED\_PAUSE}, \ \text{scope} = \text{WORD}, \ \text{strength} = \text{STRONG} \rangle.$$

14 Probability Factorization

In the factored model, the probability of an event E is decomposed into multiple components:

$$P(E) = P(\text{type}) \cdot P(\text{cue} \mid \text{type}) \cdot P(\text{scope} \mid \text{type}) \cdot P(\text{position}).$$

This factorization makes the model more expressive and modular. It allows integration of lexical, prosodic, and structural cues while maintaining probabilistic consistency. As a result, factored hidden event models provide a richer framework for speech segmentation, repair detection, and discourse modeling.

15 TextTiling

TextTiling is an unsupervised algorithm for topic segmentation. It automatically divides a document into topically coherent segments by examining changes in lexical cohesion. The main idea is that adjacent portions of text belonging to the same topic will share similar vocabulary, whereas a shift in vocabulary indicates a possible topic boundary.

The algorithm identifies positions where adjacent parts of the text become less similar and marks those positions as topic boundaries. The decision is made locally, based on neighboring blocks of text, without relying on a global view of the entire document.

16 Preprocessing and Basic Units

The document is first divided into basic units, such as:

- Fixed-size token sequences,
- Sentences,
- Paragraphs.

These units are then grouped into blocks for similarity computation. Each block typically contains w tokens (or a fixed number of sentences).

17 Method 1: Block Comparison (Cosine Similarity)

In the first method, adjacent text blocks are compared based on shared vocabulary.

Let:

- b_1, b_2 be two consecutive blocks,
- $w_{t,b}$ be the weight of term t in block b.

The similarity between b_1 and b_2 is computed using cosine similarity:

$$\mathrm{Sim}(b_1, b_2) = \frac{\sum_t w_{t,b_1} \cdot w_{t,b_2}}{\sqrt{\sum_t w_{t,b_1}^2}\sqrt{\sum_t w_{t,b_2}^2}}.$$

This represents the cosine of the angle between the two block vectors in word vector space.

Weighting Schemes

Term weights $w_{t,b}$ can be defined in different ways:

- Binary weighting (presence or absence of a term),
- Term frequency (TF),
- Other weighting schemes such as TF-IDF.

Interpretation

- High similarity indicates that the blocks discuss the same topic.
- Low similarity suggests a potential topic shift.

18 Block Size and Sliding Window

The block size is not fixed rigidly; a sliding window mechanism is often used. The window moves across the document and compares neighboring blocks. Each comparison produces a similarity score associated with the gap between blocks.

19 Method 2: Vocabulary Introduction

The second method is based on vocabulary introduction.

Intuition

A new topic typically introduces new vocabulary, whereas continuation of the same topic tends to reuse previously introduced terms.

Let:

- b_1 and b_2 be two consecutive blocks,
- Each block contain w words.

Define:

$$\text{Cohesion}(b_1, b_2) = \frac{\text{NumNewTerms}(b_1) + \text{NumNewTerms}(b_2)}{2w},$$

where NumNewTerms(b) is the number of words appearing for the first time in the document within block b.

Interpretation

- A high score indicates a vocabulary shift and suggests a topic boundary.
- A low score indicates topic continuity.

20 Thresholding and Segmentation

After computing scores for all gaps between blocks:

- The scores are smoothed to reduce noise.
- A threshold is selected.
- Gaps where the score crosses the threshold are marked as segment boundaries.

Thus, TextTiling identifies topic boundaries by detecting local minima in similarity or significant increases in vocabulary shift, resulting in a segmentation of the document into coherent topical sections.

21 Feature-Based Boundary Detection

In supervised boundary detection (sentence or topic segmentation), each potential boundary (between words or sentences) is represented by a feature vector. This leads to a classification formulation of the problem.

21.1 Problem Formulation

Each candidate boundary position is represented by a feature vector:

$$\mathbf{x} = (x_1, x_2, \ldots, x_d),$$

where each dimension corresponds to a feature extracted around that boundary. A classifier then predicts whether the boundary is present (e.g., $y = B$) or absent (e.g., $y = NB$).

22 Feature Categories

The main categories of features are:

- **Lexical features**
- **Syntactic features**
- **Discourse features**
- **Prosodic features** (for speech)

23 Types of Feature Values

23.1 Binary Features

Binary features indicate the presence or absence of a property:

$$x_j \in \{0, 1\}.$$

For example:

- Trigger word present $\Rightarrow x_j = 1$
- Otherwise $\Rightarrow x_j = 0$

Examples include punctuation marks (“.”, “?”, “!”), capitalization of the next word, or abbreviation detection.

23.2 Real-Valued Features

Real-valued features capture continuous quantities:

$$x_j \in R.$$

Examples:

- Sentence length
- Pause duration
- Similarity scores

23.3 Feature Transformation and Quantization

Some classifiers require binary or standardized inputs. Continuous features can be converted into binary indicators. For example, if a feature lies within interval $[a, b]$, we define:

$$x_j^{(a,b)} = \begin{cases} 1 & \text{if } x_j \in [a, b], \\ 0 & \text{otherwise.} \end{cases}$$

Example Binary Features

$$f_1(x, y) = \mathbf{1}[\text{word} = \text{“.”} \wedge y = B],$$

$$f_2(x, y) = \mathbf{1}[\text{next\_word\_capitalized} = \text{true} \wedge y = B],$$

$$f_3(x, y) = \mathbf{1}[\text{prev\_word\_is\_abbreviation} = \text{true} \wedge y = NB].$$

24 Lexical Features

Lexical features are derived from word identity and local word context. They often depend on final punctuation but must handle exceptions such as abbreviations (e.g., “Gov.”, “Dr.”).

Typical lexical cues include:

- n-grams before the boundary
- n-grams after the boundary

- n-grams across the boundary

For example, a boundary is unlikely after "Gov. Smith" but likely after "government." followed by a capitalized word.

24.1 Topic Boundary Similarity

For topic segmentation, let:

$$S_i = \text{sentence before boundary}, \quad S_{i+1} = \text{sentence after boundary}.$$

Cosine similarity between adjacent sentences:

$$\text{cosine}(S_i, S_{i+1}) = \frac{\sum_w tf(w, S_i)\, tf(w, S_{i+1})\, idf(w)}{\sqrt{\sum_w (tf(w, S_i) idf(w))^2} \sqrt{\sum_w (tf(w, S_{i+1}) idf(w))^2}}.$$

Similarity can also be computed over windows of multiple sentences.

24.2 Lexical Chains

Lexical chains capture semantically related word repetitions across text. A break in lexical chains may indicate a topic shift.

25 Syntactic Features

Syntactic features are derived from:

- Part-of-speech (POS) tags
- Constituency parses
- Dependency parses

Given a sequence of tags $t_1, t_2, \ldots, t_n$, features may include:

- POS n-grams before the boundary
- POS n-grams after the boundary
- POS patterns across the boundary

Syntactic features are effective for sentence segmentation but are often less useful for topic segmentation.

26 Discourse Features

Discourse features are higher-level indicators of structural change. Examples include:

- Cue phrases ("now", "by the way")
- Speaker changes
- Commercial breaks
- Dialogue acts
- Agenda transitions

Features may encode the occurrence or non-occurrence of such discourse events. For example:

$$x_e = \text{confidence score of discourse event.}$$

27 Prosodic Features (Speech)

Prosodic features encode information beyond words and are especially useful in speech segmentation.

27.1 Components

- Pitch (fundamental frequency)
- Energy (loudness)
- Timing (pause, duration)

These features:

- Are independent of word identity,
- Are robust to ASR lexical errors,
- Indicate discourse structure.

27.2 Pause Feature

Pause duration:

$$x_{\text{pause}} = \text{start}(w_{i+1}) - \text{end}(w_i).$$

Quantized version:

$$x_{\text{pause}} = \begin{cases} 1 & \text{if pause} > 0, \\ 0 & \text{otherwise.} \end{cases}$$

27.3 Pitch Feature

Pitch variation across boundary:

$$x_{\text{pitch}} = \max_{t \in w_i} \text{pitch}(t) - \min_{t \in w_{i+1}} \text{pitch}(t).$$

Pitch resets often indicate sentence or discourse boundaries.

28 Summary of Feature-Based Approach

In feature-based segmentation, each boundary candidate is represented by lexical, syntactic, discourse, and prosodic features. These features can be binary or real-valued and may be transformed or quantized depending on the classifier. A supervised model then predicts whether the candidate position corresponds to a true boundary.