Computational Linguistics

Methods of Part-of-Speech Tagging

Parameswari Krishnamurthy

Language Technologies Research Centre
IIIT-Hyderabad

param.krishna@iiit.ac.in

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

Methods of POS Tagging

- Rule-based POS tagging

Methods of POS Tagging

- Rule-based POS tagging
- Transformation-based POS tagging

Methods of POS Tagging

- Rule-based POS tagging
- Transformation-based POS tagging
- Statistical-based POS tagging

Methods of POS Tagging

- Rule-based POS tagging
- Transformation-based POS tagging
- Statistical-based POS tagging
  - Stochastic (Probabilistic) POS Tagging

Methods of POS Tagging

- Rule-based POS tagging
- Transformation-based POS tagging
- Statistical-based POS tagging
  - Stochastic (Probabilistic) POS Tagging
    - Hidden Markov Models (HMM)
    - Maximum Entropy Models
    - Conditional Random Fields (CRF)

Methods of POS Tagging

- Rule-based POS tagging
- Transformation-based POS tagging
- Statistical-based POS tagging
  - Stochastic (Probabilistic) POS Tagging
    - Hidden Markov Models (HMM)
    - Maximum Entropy Models
    - Conditional Random Fields (CRF)
- Machine Learning-based POS Tagging

Methods of POS Tagging

- Rule-based POS tagging
- Transformation-based POS tagging
- Statistical-based POS tagging
  - Stochastic (Probabilistic) POS Tagging
    - Hidden Markov Models (HMM)
    - Maximum Entropy Models
    - Conditional Random Fields (CRF)
- Machine Learning-based POS Tagging
  - Neural Networks
  - Deep Learning Models
  - Language Models (like BERT), finetuned

Methods of POS Tagging

- Rule-based POS tagging
- Transformation-based POS tagging
- Statistical-based POS tagging
  - Stochastic (Probabilistic) POS Tagging
    - Hidden Markov Models (HMM)
    - Maximum Entropy Models
    - Conditional Random Fields (CRF)
- Machine Learning-based POS Tagging
  - Neural Networks
  - Deep Learning Models
  - Language Models (like BERT), finetuned
- Hybrid Methods
  - Combination of rule-based, statistical, and machine learning approaches.
  - Example: Integration of CRFs with neural network features.

1. Rule-based Tagging

- Typically start with a dictionary of words and possible tags

1. Rule-based Tagging

- Typically start with a dictionary of words and possible tags
- Assign all possible tags to words using the dictionary

1. Rule-based Tagging

- Typically start with a dictionary of words and possible tags
- Assign all possible tags to words using the dictionary
- Write rules by hand to selectively remove tags

1. Rule-based Tagging

- Typically start with a dictionary of words and possible tags
- Assign all possible tags to words using the dictionary
- Write rules by hand to selectively remove tags
- Stop when each word has exactly one (presumably correct) tag

1. Rule-based Tagging

- Typically start with a dictionary of words and possible tags
- Assign all possible tags to words using the dictionary
- Write rules by hand to selectively remove tags
- Stop when each word has exactly one (presumably correct) tag

Example

She promised to back the bill

Example

She promised to back the bill

- She: PRP (Pronoun)
- promised: VBD (Verb, past tense) / VBN (Verb, past participle)
- to: PRP (Preposition) / TO (Infinitive marker)
- back: VB (Verb) / JJ (Adjective) / RB (Adverb) / NN (Noun)
- the: DT (Determiner)
- bill: NN (Noun) / VB (Verb)

Example

She promised to back the bill

Example

She promised to back the bill

- she: PRP

Example

She promised to back the bill

- she: PRP
- promised: ~~VBN~~,VBD

Example

She promised to back the bill

- she: PRP
- promised: ~~VBN~~,VBD
- to: ~~PRP~~, TO

Example

She promised to back the bill

- she: PRP
- promised: ~~VBN~~,VBD
- to: ~~PRP~~, TO
- back: VB, ~~JJ, RB, NN~~

Example

She promised to back the bill

- she: PRP
- promised: ~~VBN~~,VBD
- to: ~~PRP~~, TO
- back: VB, ~~JJ, RB, NN~~
- the: DT

Example

She promised to back the bill

- she: PRP
- promised: ~~VBN~~,VBD
- to: ~~PRP~~, TO
- back: VB, ~~JJ, RB, NN~~
- the: DT
- bill: NN, ~~VB~~

1. **First Stage:**
   - Utilizes a dictionary to assign each word a list of potential parts-of-speech.

- **First Stage:**
  - Utilizes a dictionary to assign each word a list of potential parts-of-speech.
- **Second Stage:**
  - Applies large lists of hand-written disambiguation rules to narrow down the list to a single part-of-speech for each word.

EngCG ENGTWOL Tagger

- A rule-based POS tagger designed for English and other languages.

EngCG ENGTWOL Tagger

- A rule-based POS tagger designed for English and other languages.
- Richer Dictionary includes morphological and syntactic features (e.g., subcategorization frames) as well as possible POS.

EngCG ENGTWOL Tagger

- A rule-based POS tagger designed for English and other languages.
- Richer Dictionary includes morphological and syntactic features (e.g., subcategorization frames) as well as possible POS.
- Uses two-level morphological analysis on input and returns all possible POS.

EngCG ENGTWOL Tagger

- A rule-based POS tagger designed for English and other languages.
- Richer Dictionary includes morphological and syntactic features (e.g., subcategorization frames) as well as possible POS.
- Uses two-level morphological analysis on input and returns all possible POS.
- Apply negative constraints ($>$3744) to rule out incorrect POS.

EngCG ENGTWOL Tagger

- A rule-based POS tagger designed for English and other languages.
- Richer Dictionary includes morphological and syntactic features (e.g., subcategorization frames) as well as possible POS.
- Uses two-level morphological analysis on input and returns all possible POS.
- Apply negative constraints ($>$3744) to rule out incorrect POS.

Rule-based POS Tagging: Pros and Cons

Rule-based POS Tagging: Pros and Cons

Pros:

- **Transparency:** Easy to understand and interpret.
- **Customization:** Can be tailored to specific languages or domains.
- **No Training Data Required:** Doesn't require a large annotated corpus.

Cons:

- **Limited Coverage:** May miss out on complex linguistic phenomena.
- **Scalability:** Difficult to maintain as the number of rules increases.
- **Performance:** Generally less accurate than statistical or machine learning methods.

Transformation-based Tagging

Transformation-based Tagging (TBL)

- **Introduced by:** Eric Brill in 1992.

Transformation-based Tagging

Transformation-based Tagging (TBL)

- **Introduced by:** Eric Brill in 1992.
- **Objective:** Develops a rule-based model that iteratively refines an initial state through transformations.

Transformation-based Tagging

Transformation-based Tagging (TBL)

- **Introduced by:** Eric Brill in 1992.
- **Objective:** Develops a rule-based model that iteratively refines an initial state through transformations.
- **Key Components:**

Transformation-based Tagging

Transformation-based Tagging (TBL)

- **Introduced by:** Eric Brill in 1992.
- **Objective:** Develops a rule-based model that iteratively refines an initial state through transformations.
- **Key Components:**
  - **Initial State:** Assigns a naive label, often using a simple method like the most frequent tag.

Transformation-based Tagging

Transformation-based Tagging (TBL)

- **Introduced by:** Eric Brill in 1992.
- **Objective:** Develops a rule-based model that iteratively refines an initial state through transformations.
- **Key Components:**
  - **Initial State:** Assigns a naive label, often using a simple method like the most frequent tag.
  - **Transformations:** Rules that correct errors in the current tagging based on context.

Transformation-based Tagging

Transformation-based Tagging (TBL)

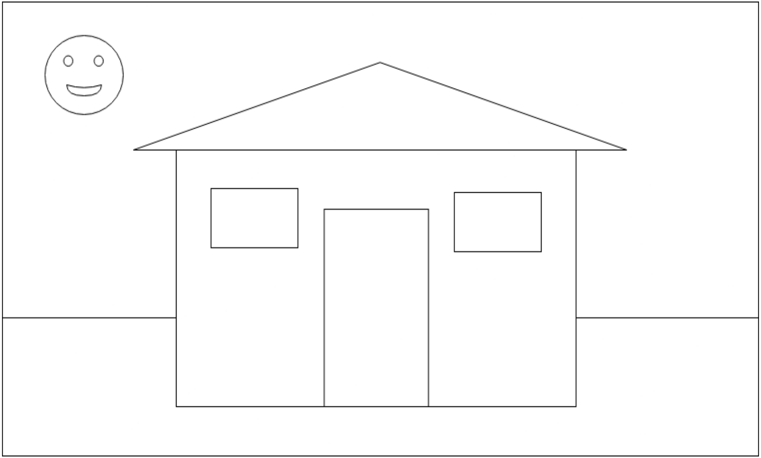
- **Introduced by:** Eric Brill in 1992.
- **Objective:** Develops a rule-based model that iteratively refines an initial state through transformations.
- **Key Components:**
  - **Initial State:** Assigns a naive label, often using a simple method like the most frequent tag.
  - **Transformations:** Rules that correct errors in the current tagging based on context.
  - **Learning Process:**
    - Identify errors in current tagging.
    - Create a transformation rule to correct these errors.
    - Apply the rule and iterate.

Transformation-based Tagging

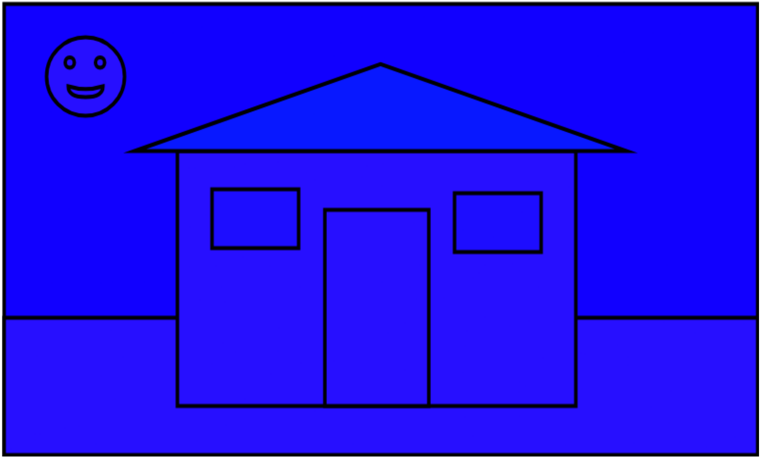
Transformation-based Tagging (TBL)

- **Introduced by:** Eric Brill in 1992.
- **Objective:** Develops a rule-based model that iteratively refines an initial state through transformations.
- **Key Components:**
  - **Initial State:** Assigns a naive label, often using a simple method like the most frequent tag.
  - **Transformations:** Rules that correct errors in the current tagging based on context.
  - **Learning Process:**
    - Identify errors in current tagging.
    - Create a transformation rule to correct these errors.
    - Apply the rule and iterate.
  - **Final Output:** A sequence of rules applied to improve tagging accuracy.

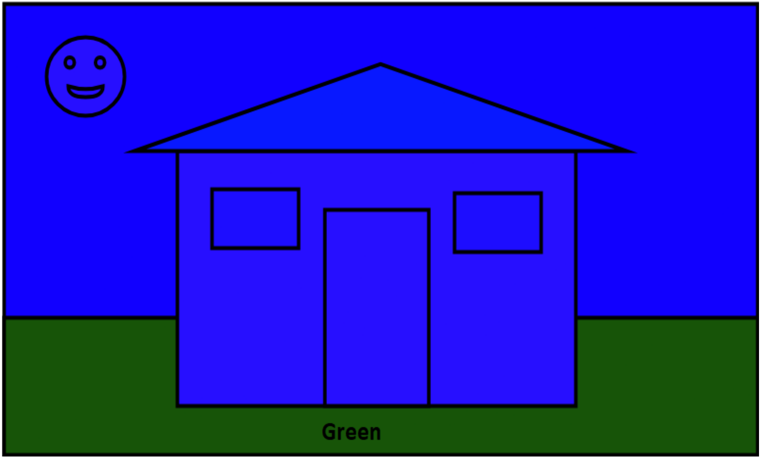
TBL as Painting Analogy



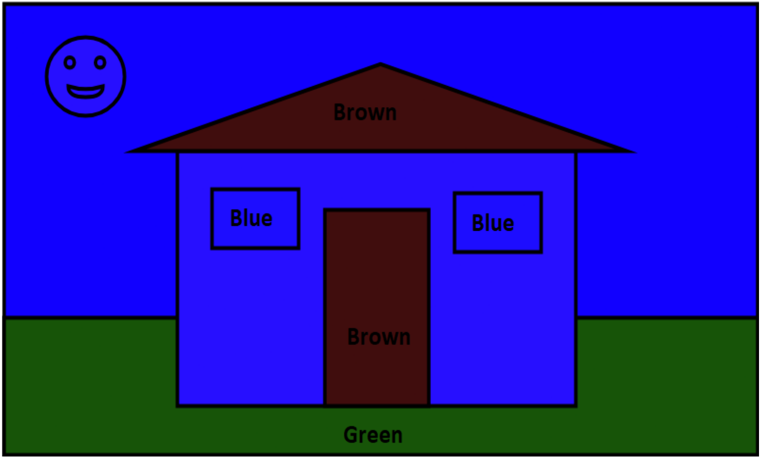
TBL as Painting Analogy



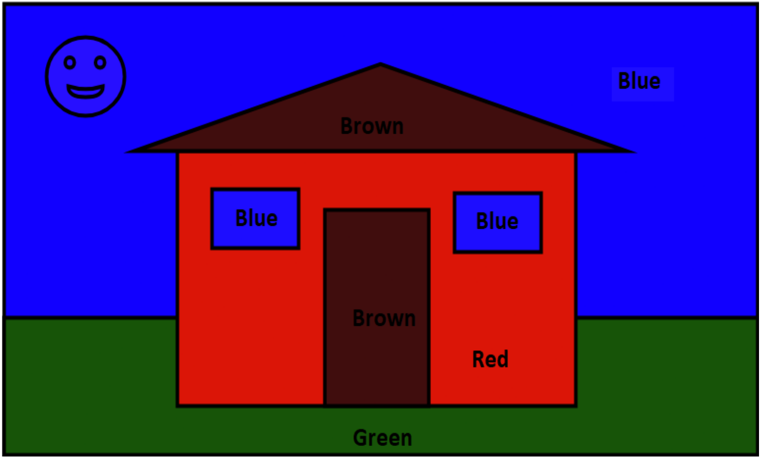
TBL as Painting Analogy



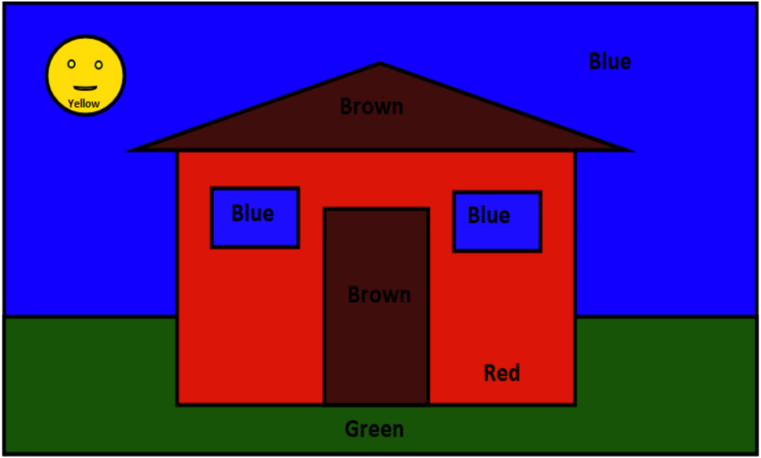
TBL as Painting Analogy



TBL as Painting Analogy



TBL as Painting Analogy



Transformation Rules

Change tag `a` to `b` when:

Transformation Rules

Change tag `a` to `b` when:

- **Internal evidence:** Based on morphology.

Transformation Rules

Change tag `a` to `b` when:

- **Internal evidence:** Based on morphology.
- **Contextual evidence:**
  - One or more of the preceding/following words has a specific tag.

Transformation Rules

Change tag `a` to `b` when:

- **Internal evidence:** Based on morphology.
- **Contextual evidence:**
  - One or more of the preceding/following words has a specific tag.
  - One or more of the preceding/following words is a specific word.

Transformation Rules

Change tag `a` to `b` when:

- **Internal evidence:** Based on morphology.
- **Contextual evidence:**
  - One or more of the preceding/following words has a specific tag.
  - One or more of the preceding/following words is a specific word.
  - One or more of the preceding/following words has a certain form.

Transformation Rules

Change tag `a` to `b` when:

- **Internal evidence:** Based on morphology.
- **Contextual evidence:**
  - One or more of the preceding/following words has a specific tag.
  - One or more of the preceding/following words is a specific word.
  - One or more of the preceding/following words has a certain form.
- **Order of rules is important.**

Transformation Rules

Change tag `a` to `b` when:

- **Internal evidence:** Based on morphology.
- **Contextual evidence:**
  - One or more of the preceding/following words has a specific tag.
  - One or more of the preceding/following words is a specific word.
  - One or more of the preceding/following words has a certain form.
- **Order of rules is important.**
- **Cascading effects:**
  - Rules can change a correct tag into an incorrect tag.
  - Another rule might be required to correct that "mistake."

Transformation-based Tagging: Examples

Examples of Transformation Rules:

- **Rule 1:**
  If a word is currently tagged `NN`, and has a suffix of length 1 which consists of the letter `'s'`, change its tag to `NNS`.

Transformation-based Tagging: Examples

Examples of Transformation Rules:

- **Rule 1:**
  If a word is currently tagged `NN`, and has a suffix of length 1 which consists of the letter `'s'`, change its tag to `NNS`.
- **Rule 2:**
  If a word has a suffix of length 2 consisting of the letter sequence `'ly'`, change its tag to `RB` (regardless of the initial tag).

Transformation-based Tagging: Examples

Examples of Transformation Rules:

- **Rule 1:**
  If a word is currently tagged `NN`, and has a suffix of length 1 which consists of the letter `'s'`, change its tag to `NNS`.
- **Rule 2:**
  If a word has a suffix of length 2 consisting of the letter sequence `'ly'`, change its tag to `RB` (regardless of the initial tag).
- **Rule 3:**
  Change `VBN` to `VBD` if the previous word is tagged as `NN`.

Transformation-based Tagging: Examples

Examples of Transformation Rules:

- **Rule 1:**
  If a word is currently tagged `NN`, and has a suffix of length 1 which consists of the letter `'s'`, change its tag to `NNS`.
- **Rule 2:**
  If a word has a suffix of length 2 consisting of the letter sequence `'ly'`, change its tag to `RB` (regardless of the initial tag).
- **Rule 3:**
  Change `VBN` to `VBD` if the previous word is tagged as `NN`.
- **Rule 4:**
  Change `VBD` to `VBN` if the previous word is `'by'`.

Transformation-based Tagging: Example Sequence

Example after Lexical Lookup:

Transformation-based Tagging: Example Sequence

Example after Lexical Lookup:

- Booth/NP killed/VBN Abraham/NP Lincoln/NP
- Abraham/NP Lincoln/NP was/AUX shot/VBD by/BY Booth/NP
- He/PPS witnessed/VBD Lincoln/NP killed/VBN by/BY Booth/NP

After Applying Contextual Rule: `vbn` $\rightarrow$ `vbd PREVTAG np`

Transformation-based Tagging: Example Sequence

Example after Lexical Lookup:

- Booth/NP killed/VBN Abraham/NP Lincoln/NP
- Abraham/NP Lincoln/NP was/AUX shot/VBD by/BY Booth/NP
- He/PPS witnessed/VBD Lincoln/NP killed/VBN by/BY Booth/NP

After Applying Contextual Rule: `vbn` $\rightarrow$ `vbd PREVTAG np`

- Booth/NP killed/**VBD** Abraham/NP Lincoln/NP
- Abraham/NP Lincoln/NP was/AUX shot/**VBD** by/BY Booth/NP
- He/PPS witnessed/**VBD** Lincoln/NP killed/VBD by/BY Booth/NP

After Applying Contextual Rule: `vbd` $\rightarrow$ `vbn NEXTWORD by`

Transformation-based Tagging: Example Sequence

Example after Lexical Lookup:

- Booth/NP killed/VBN Abraham/NP Lincoln/NP
- Abraham/NP Lincoln/NP was/AUX shot/VBD by/BY Booth/NP
- He/PPS witnessed/VBD Lincoln/NP killed/VBN by/BY Booth/NP

After Applying Contextual Rule: `vbn` $\rightarrow$ `vbd PREVTAG np`

- Booth/NP killed/**VBD** Abraham/NP Lincoln/NP
- Abraham/NP Lincoln/NP was/AUX shot/**VBD** by/BY Booth/NP
- He/PPS witnessed/**VBD** Lincoln/NP killed/VBD by/BY Booth/NP

After Applying Contextual Rule: `vbd` $\rightarrow$ `vbn NEXTWORD by`

- Booth/NP killed/**VBD** Abraham/NP Lincoln/NP
- Abraham/NP Lincoln/NP was/AUX shot/**VBN** by/BY Booth/NP
- He/PPS witnessed/**VBD** Lincoln/NP killed/VBN by/BY Booth/NP

Transformation-Based (Brill) Tagging

Combines Rule-based and Stochastic Tagging:

Transformation-Based (Brill) Tagging

Combines Rule-based and Stochastic Tagging:

- **Rule-based:** Uses rules to specify tags in a certain environment.

Transformation-Based (Brill) Tagging

Combines Rule-based and Stochastic Tagging:

- **Rule-based:** Uses rules to specify tags in a certain environment.
- **Stochastic approach:** Utilizes a tagged corpus to find the best-performing rules.

Transformation-Based (Brill) Tagging

Combines Rule-based and Stochastic Tagging:

- **Rule-based:** Uses rules to specify tags in a certain environment.
- **Stochastic approach:** Utilizes a tagged corpus to find the best-performing rules.

Key Features:

- Rules are learned from data.

Input:

- **Tagged Corpus:** Provides the training data for learning rules.
- **Dictionary:** Contains the most frequent tags associated with words.

Brill Tagger: Process Overview

Initial Tagging:

- **Step 1:** The Brill Tagger starts with a base tagging method that assigns a preliminary tag to each word.

Brill Tagger: Process Overview

Initial Tagging:

- **Step 1:** The Brill Tagger starts with a base tagging method that assigns a preliminary tag to each word.
- **Method:** Often uses a simple statistical model, such as the most frequent tag for each word.

Brill Tagger: Process Overview

Initial Tagging:

- **Step 1:** The Brill Tagger starts with a base tagging method that assigns a preliminary tag to each word.
- **Method:** Often uses a simple statistical model, such as the most frequent tag for each word.

Rule Learning:

- **Step 2:** Iteratively applies a set of transformation rules to refine the initial tags.

Brill Tagger: Process Overview

Initial Tagging:

- **Step 1:** The Brill Tagger starts with a base tagging method that assigns a preliminary tag to each word.
- **Method:** Often uses a simple statistical model, such as the most frequent tag for each word.

Rule Learning:

- **Step 2:** Iteratively applies a set of transformation rules to refine the initial tags.
- **Transformation Rules:** Each rule consists of a condition and an action.

Brill Tagger: Process Overview

Initial Tagging:

- **Step 1:** The Brill Tagger starts with a base tagging method that assigns a preliminary tag to each word.
- **Method:** Often uses a simple statistical model, such as the most frequent tag for each word.

Rule Learning:

- **Step 2:** Iteratively applies a set of transformation rules to refine the initial tags.
- **Transformation Rules:** Each rule consists of a condition and an action. For example:
  - *Condition:* If a word is tagged as a noun and follows a preposition.
  - *Action:* Change the tag to a verb.

Brill Tagger: Process Overview

Evaluation and Application:

- **Step 3:** After each rule is applied, the tagging accuracy is evaluated.

Brill Tagger: Process Overview

Evaluation and Application:

- **Step 3:** After each rule is applied, the tagging accuracy is evaluated.
- **Selection:** The rule that most improves accuracy is retained.
- **Iteration:** Process is repeated until a stopping criterion is met (e.g., no further improvements).

Brill Tagger: Process Overview

Evaluation and Application:

- **Step 3:** After each rule is applied, the tagging accuracy is evaluated.
- **Selection:** The rule that most improves accuracy is retained.
- **Iteration:** Process is repeated until a stopping criterion is met (e.g., no further improvements).

Final Output:

- **Result:** A sequence of ordered transformation rules applied to new data to produce accurate POS tags.

TBL Tagging Algorithm

Step 1:
Label every word with the most likely tag (from dictionary).

TBL Tagging Algorithm

Step 1:
Label every word with the most likely tag (from dictionary).

Step 2:
Check every possible transformation and select the one which most improves tag accuracy (compared to the gold standard).

TBL Tagging Algorithm

Step 1:
Label every word with the most likely tag (from dictionary).

Step 2:
Check every possible transformation and select the one which most improves tag accuracy (compared to the gold standard).

Step 3:
Re-tag the corpus by applying this rule, and add the rule to the end of the rule set.

TBL Tagging Algorithm

Step 1:
Label every word with the most likely tag (from dictionary).

Step 2:
Check every possible transformation and select the one which most improves tag accuracy (compared to the gold standard).

Step 3:
Re-tag the corpus by applying this rule, and add the rule to the end of the rule set.

Repeat Steps 2-3
Continue until some stopping criterion is reached, e.g., X% correct with respect to the training corpus.

TBL Tagging Algorithm

Step 1:
Label every word with the most likely tag (from dictionary).

Step 2:
Check every possible transformation and select the one which most improves tag accuracy (compared to the gold standard).

Step 3:
Re-tag the corpus by applying this rule, and add the rule to the end of the rule set.

Repeat Steps 2-3
Continue until some stopping criterion is reached, e.g., X% correct with respect to the training corpus.

Result:
An ordered set of transformation rules to use on new data that is tagged only with the most likely POS tags.

TBL Tagging Algorithm

Step 1:
Label every word with the most likely tag (from dictionary).

Step 2:
Check every possible transformation and select the one which most improves tag accuracy (compared to the gold standard).

Step 3:
Re-tag the corpus by applying this rule, and add the rule to the end of the rule set.

Repeat Steps 2-3
Continue until some stopping criterion is reached, e.g., X% correct with respect to the training corpus.

Result:
An ordered set of transformation rules to use on new data that is tagged only with the most likely POS tags.

Transformation-Based Tagging: Example

Step 1: Label every word with its most-likely tag

- **Example:**

Transformation-Based Tagging: Example

Step 1: Label every word with its most-likely tag

- **Example:**
  The word "race" occurs in the Brown corpus with the following probabilities:
  `P(NN|race) = 0.98`
  `P(VB|race) = 0.02`
- Initial tagging:
  `... is/VBZ expected/VBN to/TO race/NN tomorrow/NN`

Step 2: Apply Transformation Rule

- **Rule:**

Transformation-Based Tagging: Example

Step 1: Label every word with its most-likely tag

- **Example:**
  The word "race" occurs in the Brown corpus with the following probabilities:
  `P(NN|race) = 0.98`
  `P(VB|race) = 0.02`
- Initial tagging:
  `... is/VBZ expected/VBN to/TO race/NN tomorrow/NN`

Step 2: Apply Transformation Rule

- **Rule:**
  "Change `NN` to `VB` when the previous tag is `TO`."
- Tagging after applying rule:
  `... is/VBZ expected/VBN to/TO race/VB tomorrow/NN`

Explanation:

- The word "race" was initially tagged as a noun (`NN`) because it is more commonly used as a noun (with a probability of 0.98). However, based on the context provided by the preceding word "to" (tagged as `TO`), the rule changes the tag to `VB` (verb), which makes more sense in this context.

3. Stochastic (Probabilistic) Tagging

Overview:

- Uses statistical methods to predict POS tags based on probabilities.

3. Stochastic (Probabilistic) Tagging

Overview:

- Uses statistical methods to predict POS tags based on probabilities.
- Models the likelihood of tag sequences and word-tag pairs.

Key Concepts:

- **Probability Model:** Estimates the probability of a sequence of tags given a sequence of words.

3. Stochastic (Probabilistic) Tagging

Overview:

- Uses statistical methods to predict POS tags based on probabilities.
- Models the likelihood of tag sequences and word-tag pairs.

Key Concepts:

- **Probability Model:** Estimates the probability of a sequence of tags given a sequence of words.
- **Training Data:** A tagged corpus is used to estimate probabilities.

3. Stochastic (Probabilistic) Tagging

Overview:

- Uses statistical methods to predict POS tags based on probabilities.
- Models the likelihood of tag sequences and word-tag pairs.

Key Concepts:

- **Probability Model:** Estimates the probability of a sequence of tags given a sequence of words.
- **Training Data:** A tagged corpus is used to estimate probabilities.
- **Tag Sequences:** Determines the most likely sequence of tags for a given sequence of words.

3. Stochastic (Probabilistic) Tagging

Overview:

- Uses statistical methods to predict POS tags based on probabilities.
- Models the likelihood of tag sequences and word-tag pairs.

Key Concepts:

- **Probability Model:** Estimates the probability of a sequence of tags given a sequence of words.
- **Training Data:** A tagged corpus is used to estimate probabilities.
- **Tag Sequences:** Determines the most likely sequence of tags for a given sequence of words.

How do they work?

Training a POS Tagger: Key Points

- **Training Process:**
  - The tagger must be “trained” on a corpus of tagged data.
  - Training involves learning tagging rules automatically from this data.

How do they work?

Training a POS Tagger: Key Points

- **Training Process:**
  - The tagger must be "trained" on a corpus of tagged data.
  - Training involves learning tagging rules automatically from this data.
- **Training Corpus:**
  - Typically, a small "training corpus" is used.
  - The corpus is hand-tagged with POS tags to provide a reference.

How do they work?

Training a POS Tagger: Key Points

- **Training Process:**
  - The tagger must be "trained" on a corpus of tagged data.
  - Training involves learning tagging rules automatically from this data.
- **Training Corpus:**
  - Typically, a small "training corpus" is used.
  - The corpus is hand-tagged with POS tags to provide a reference.
- **Learning Tagging Rules:**
  - Rules are defined to identify the most likely sequence of tags.
  - Rules are based on:

How do they work?

Training a POS Tagger: Key Points

- **Training Process:**
  - The tagger must be "trained" on a corpus of tagged data.
  - Training involves learning tagging rules automatically from this data.
- **Training Corpus:**
  - Typically, a small "training corpus" is used.
  - The corpus is hand-tagged with POS tags to provide a reference.
- **Learning Tagging Rules:**
  - Rules are defined to identify the most likely sequence of tags.
  - Rules are based on:
    - **Internal Evidence:** Morphological features of words (e.g., suffixes).

How do they work?

Training a POS Tagger: Key Points

- **Training Process:**
  - The tagger must be "trained" on a corpus of tagged data.
  - Training involves learning tagging rules automatically from this data.
- **Training Corpus:**
  - Typically, a small "training corpus" is used.
  - The corpus is hand-tagged with POS tags to provide a reference.
- **Learning Tagging Rules:**
  - Rules are defined to identify the most likely sequence of tags.
  - Rules are based on:
    - **Internal Evidence:** Morphological features of words (e.g., suffixes).
    - **External Evidence:** Context provided by surrounding words.

How do they work?

Training a POS Tagger: Key Points

- **Training Process:**
  - The tagger must be “trained” on a corpus of tagged data.
  - Training involves learning tagging rules automatically from this data.
- **Training Corpus:**
  - Typically, a small “training corpus” is used.
  - The corpus is hand-tagged with POS tags to provide a reference.
- **Learning Tagging Rules:**
  - Rules are defined to identify the most likely sequence of tags.
  - Rules are based on:
    - **Internal Evidence:** Morphological features of words (e.g., suffixes).
    - **External Evidence:** Context provided by surrounding words.
    - **Probabilities:** Statistical models estimating the likelihood of tag sequences.

How do they work?

Training a POS Tagger: Key Points

- **Training Process:**
  - The tagger must be “trained” on a corpus of tagged data.
  - Training involves learning tagging rules automatically from this data.
- **Training Corpus:**
  - Typically, a small “training corpus” is used.
  - The corpus is hand-tagged with POS tags to provide a reference.
- **Learning Tagging Rules:**
  - Rules are defined to identify the most likely sequence of tags.
  - Rules are based on:
    - **Internal Evidence:** Morphological features of words (e.g., suffixes).
    - **External Evidence:** Context provided by surrounding words.
    - **Probabilities:** Statistical models estimating the likelihood of tag sequences.
- Two types of Probabilities
  - Individual word Probabilities

How do they work?

Training a POS Tagger: Key Points

- **Training Process:**
  - The tagger must be "trained" on a corpus of tagged data.
  - Training involves learning tagging rules automatically from this data.
- **Training Corpus:**
  - Typically, a small "training corpus" is used.
  - The corpus is hand-tagged with POS tags to provide a reference.
- **Learning Tagging Rules:**
  - Rules are defined to identify the most likely sequence of tags.
  - Rules are based on:
    - **Internal Evidence:** Morphological features of words (e.g., suffixes).
    - **External Evidence:** Context provided by surrounding words.
    - **Probabilities:** Statistical models estimating the likelihood of tag sequences.
- Two types of Probabilities
  - Individual word Probabilities
  - Tag Sequence Probabilities

Learning Individual Word Probabilities

Individual Word Probabilities:

- The probability that a given tag t is appropriate for a given word w is estimated from the training corpus.
- This probability helps in determining the likelihood of each possible tag for a word.

Learning Individual Word Probabilities

Individual Word Probabilities:

- The probability that a given tag t is appropriate for a given word w is estimated from the training corpus.
- This probability helps in determining the likelihood of each possible tag for a word.

Formula for Learning Individual Word Probabilities:

- The probability $P(t \mid w)$ is the probability that a given tag t is appropriate for a given word w.

Learning Individual Word Probabilities

Individual Word Probabilities:

- The probability that a given tag t is appropriate for a given word w is estimated from the training corpus.
- This probability helps in determining the likelihood of each possible tag for a word.

Formula for Learning Individual Word Probabilities:

- The probability $P(t \mid w)$ is the probability that a given tag t is appropriate for a given word w.
- It is calculated using:

$$P(t \mid w) = \frac{f(t, w)}{f(w)}$$

Where:

- $f(t, w)$ is the frequency of the word w occurring with the tag t.
- $f(w)$ is the total frequency of the word w in the corpus.

Example:

- Consider the word "run" which occurs 4800 times in the training corpus.
  - 3600 times as a verb
  - 1200 times as a noun
- The probability of "run" being a verb is calculated as:

Example:

- Consider the word "run" which occurs 4800 times in the training corpus.
  - 3600 times as a verb
  - 1200 times as a noun
- The probability of "run" being a verb is calculated as:

$$P(\text{verb} \mid \text{run}) = \frac{\text{Number of occurrences of "run" as a verb}}{\text{Total number of occurrences of "run"}} = \frac{3600}{4800} = 0.75$$

Example:

- Consider the word "run" which occurs 4800 times in the training corpus.
  - 3600 times as a verb
  - 1200 times as a noun
- The probability of "run" being a verb is calculated as:

$$P(\text{verb} \mid \text{run}) = \frac{\text{Number of occurrences of "run" as a verb}}{\text{Total number of occurrences of "run"}} = \frac{3600}{4800} = 0.75$$

Interpretation:

- $P(\text{verb} \mid \text{run}) = 0.75$ indicates that there is a 75% probability that the word "run" is used as a verb.

Tag Sequence Probability

Tag Sequence Probability:

- The probability that a given tag sequence $t_1, t_2, \ldots, t_n$ is appropriate for a given word sequence $w_1, w_2, \ldots, w_n$ is denoted as:

Tag Sequence Probability

Tag Sequence Probability:

- The probability that a given tag sequence $t_1, t_2, \ldots, t_n$ is appropriate for a given word sequence $w_1, w_2, \ldots, w_n$ is denoted as:

$$P(t_1, t_2, \ldots, t_n \mid w_1, w_2, \ldots, w_n)$$

Tag Sequence Probability

Tag Sequence Probability:

- The probability that a given tag sequence $t_1, t_2, \ldots, t_n$ is appropriate for a given word sequence $w_1, w_2, \ldots, w_n$ is denoted as:

$$P(t_1, t_2, \ldots, t_n \mid w_1, w_2, \ldots, w_n)$$

Challenge:

- Calculating this probability directly for the entire sequence is complex.
- Instead, it's often simplified using the chain rule:

$$P(t_1, t_2, t_3, \ldots \mid w_1, w_2, w_3, \ldots) = P(t_1) \times P(t_2 \mid t_1) \times P(t_3 \mid t_1, t_2) \times \cdots$$

Explanation:

- The joint probability is decomposed into a product of conditional probabilities.
- This approach reduces complexity by breaking down the problem into manageable parts.

Subsequence Models for Tag Probability

Using Subsequences:

- Direct computation of full sequence probabilities is complex.
- Simplified models using smaller subsequences (2 or 3 tags) are more tractable.

Subsequence Models for Tag Probability

Using Subsequences:

- Direct computation of full sequence probabilities is complex.
- Simplified models using smaller subsequences (2 or 3 tags) are more tractable.
- **Bigram Model:**
  - Uses a sequence of 2 tags.
  - Probability of a tag sequence t_1, t_2 is:

$$P(t_1, t_2) = P(t_2 \mid t_1)$$

Subsequence Models for Tag Probability

Using Subsequences:

- Direct computation of full sequence probabilities is complex.
- Simplified models using smaller subsequences (2 or 3 tags) are more tractable.
- **Bigram Model:**
  - Uses a sequence of 2 tags.
  - Probability of a tag sequence t_1, t_2 is:

$$P(t_1, t_2) = P(t_2 \mid t_1)$$

- **Trigram Model:**
  - Uses a sequence of 3 tags.
  - Probability of a tag sequence t_1, t_2, t_3 is:

$$P(t_1, t_2, t_3) = P(t_2 \mid t_1) \times P(t_3 \mid t_2)$$

Subsequence Models for Tag Probability

Using Subsequences:

- Direct computation of full sequence probabilities is complex.
- Simplified models using smaller subsequences (2 or 3 tags) are more tractable.
- **Bigram Model:**
  - Uses a sequence of 2 tags.
  - Probability of a tag sequence t_1, t_2 is:

$$P(t_1, t_2) = P(t_2 \mid t_1)$$

- **Trigram Model:**
  - Uses a sequence of 3 tags.
  - Probability of a tag sequence t_1, t_2, t_3 is:

$$P(t_1, t_2, t_3) = P(t_2 \mid t_1) \times P(t_3 \mid t_2)$$

- **N-gram Model:**
  - Generalizes to sequences of N tags.
  - Probability of a tag sequence $t_1, t_2, \ldots, t_N$ is:

$$P(t_1, t_2, \ldots, t_N) = \prod_{i=2}^{N} P(t_i \mid t_{i-1}, \ldots, t_{i-N+1})$$

Common Models:

- **Hidden Markov Model (HMM):**
  - Models sequences of tags (states) and words (observations).
  - Uses transition probabilities (tag-to-tag) and emission probabilities (word-to-tag).

Common Models:

- **Hidden Markov Model (HMM):**
  - Models sequences of tags (states) and words (observations).
  - Uses transition probabilities (tag-to-tag) and emission probabilities (word-to-tag).
- **Maximum Entropy Models:**
  - Uses a probabilistic model to handle multiple features and constraints.

Common Models:

- **Hidden Markov Model (HMM):**
  - Models sequences of tags (states) and words (observations).
  - Uses transition probabilities (tag-to-tag) and emission probabilities (word-to-tag).
- **Maximum Entropy Models:**
  - Uses a probabilistic model to handle multiple features and constraints.
- **Conditional Random Fields (CRF):**
  - Models the conditional probability of a tag sequence given the observed data.
  - Handles dependencies between tags more flexibly.

Common Models:

- **Hidden Markov Model (HMM):**
  - Models sequences of tags (states) and words (observations).
  - Uses transition probabilities (tag-to-tag) and emission probabilities (word-to-tag).
- **Maximum Entropy Models:**
  - Uses a probabilistic model to handle multiple features and constraints.
- **Conditional Random Fields (CRF):**
  - Models the conditional probability of a tag sequence given the observed data.
  - Handles dependencies between tags more flexibly.

POS Evaluation

Evaluating POS (Part-of-Speech) tagging systems involves measuring how well the system's output matches the gold standard (human-defined) labels.
We will explore:

- Confusion Matrix

POS Evaluation

Evaluating POS (Part-of-Speech) tagging systems involves measuring how well the system's output matches the gold standard (human-defined) labels.
We will explore:

- Confusion Matrix
- Accuracy

POS Evaluation

Evaluating POS (Part-of-Speech) tagging systems involves measuring how well the system's output matches the gold standard (human-defined) labels.
We will explore:

- Confusion Matrix
- Accuracy
- Precision and Recall

POS Evaluation

Evaluating POS (Part-of-Speech) tagging systems involves measuring how well the system's output matches the gold standard (human-defined) labels.
We will explore:

- Confusion Matrix
- Accuracy
- Precision and Recall
- F-Measure

POS Evaluation

Evaluating POS (Part-of-Speech) tagging systems involves measuring how well the system's output matches the gold standard (human-defined) labels.
We will explore:

- Confusion Matrix
- Accuracy
- Precision and Recall
- F-Measure

Confusion Matrix

A confusion matrix visualizes how a tagging system performs relative to gold labels. It includes:

- **True Positives (TP)**: Correctly identified positive cases

Confusion Matrix

A confusion matrix visualizes how a tagging system performs relative to gold labels. It includes:

- **True Positives (TP)**: Correctly identified positive cases
- **False Positives (FP)**: Incorrectly identified as positive

Confusion Matrix

A confusion matrix visualizes how a tagging system performs relative to gold labels. It includes:

- **True Positives (TP)**: Correctly identified positive cases
- **False Positives (FP)**: Incorrectly identified as positive
- **True Negatives (TN)**: Correctly identified negative cases

Confusion Matrix

A confusion matrix visualizes how a tagging system performs relative to gold labels. It includes:

- **True Positives (TP)**: Correctly identified positive cases
- **False Positives (FP)**: Incorrectly identified as positive
- **True Negatives (TN)**: Correctly identified negative cases
- **False Negatives (FN)**: Incorrectly identified as negative

Confusion Matrix

A confusion matrix visualizes how a tagging system performs relative to gold labels. It includes:

- **True Positives (TP)**: Correctly identified positive cases
- **False Positives (FP)**: Incorrectly identified as positive
- **True Negatives (TN)**: Correctly identified negative cases
- **False Negatives (FN)**: Incorrectly identified as negative

	Gold Positive	Gold Negative
System Positive	TP	FP
System Negative	FN	TN

Accuracy

Accuracy measures the overall correctness of the tagging system. It is defined as:

Accuracy

Accuracy measures the overall correctness of the tagging system. It is defined as:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}$$

Accuracy

Accuracy measures the overall correctness of the tagging system. It is defined as:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}$$

Accuracy can be misleading in cases of class imbalance, where one class is much more frequent than the other.

Precision and Recall

Precision measures the accuracy of positive predictions:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Precision and Recall

Precision measures the accuracy of positive predictions:

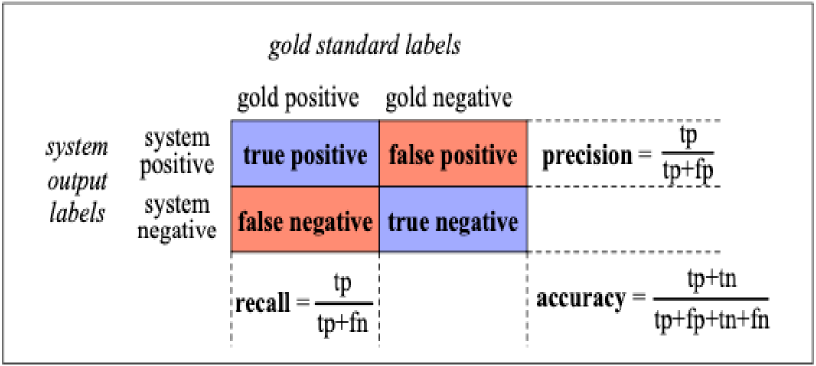
$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Recall measures the ability to find all relevant positive instances:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Both precision and recall are important metrics, especially in cases of unbalanced datasets.

Precision, Recall and Accuracy



F-Measure

The F-measure combines precision and recall into a single metric. It is defined as:

F-Measure

The F-measure combines precision and recall into a single metric. It is defined as:

$$F_\beta = \frac{(1 + \beta^2) \cdot \text{Precision} \cdot \text{Recall}}{\beta^2 \cdot \text{Precision} + \text{Recall}}$$

F-Measure

The F-measure combines precision and recall into a single metric. It is defined as:

$$F_\beta = \frac{(1+\beta^2)\cdot \text{Precision} \cdot \text{Recall}}{\beta^2 \cdot \text{Precision} + \text{Recall}}$$

For $\beta = 1$, it simplifies to the F1 score:

$$F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

F-Measure

The F-measure combines precision and recall into a single metric. It is defined as:

$$F_\beta = \frac{(1+\beta^2)\cdot \text{Precision} \cdot \text{Recall}}{\beta^2 \cdot \text{Precision} + \text{Recall}}$$

For $\beta = 1$, it simplifies to the F1 score:

$$F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

The F-measure balances precision and recall, with β adjusting the importance of recall versus precision.