

דוגמה מתקדמת - תבנית אקדמית עברית

Advanced Example - Hebrew Academic Template v5.0

ד"ר סgal יoram

כל הזכויות שמורות © Dr. Segal Yoram

November 2025

גרסה 0.5 - מתקדמת

תוכן העניינים

| | | |
|----|-------------------------------------|-------|
| 3 | מבוא מתקדם: | 0.1 |
| 3 | סקירת ספרות מקיפה: | 0.1.1 |
| 3 | טבלאות מורכבות וمتתקדמות: | 0.2 |
| 3 | טבלת ביצועים מפורשת: | 0.2.1 |
| 4 | טבלת השוואת משאבים: | 0.2.2 |
| 4 | דוגמאות קוד מרובות: | 0.3 |
| 4 | שימוש מלא של רשות: | 0.3.1 |
| 5 | עיבוד נתונים מתקדם: | 0.3.2 |
| 6 | הפניות צולבות מתקדמות: | 0.4 |
| 6 | הפניות לטבלאות ואיורים: | 0.4.1 |
| 6 | מתמטיקה מתקדמת עם עברית: | 0.5 |
| 6 | אופטימיזציה וגרדיינטיים: | 0.5.1 |
| 7 | מטריצות ווקטוריים: | 0.5.2 |
| 7 | ביבליוגרפיה מתקדמת: | 0.6 |
| 7 | ציטוטים מרובים ומורכבים: | 0.6.1 |
| 7 | סוגי ציטוטים שונים: | 0.6.2 |
| 8 | תכונות מתקדמות נוספות: | 0.7 |
| 8 | טיפול במספרים מורכבים: | 0.7.1 |
| 8 | שילוב תוכן מרכיב: | 0.7.2 |
| 9 | סיכום ומסקנות מתקדמות: | 0.8 |
| 10 | מקורות בעברית | 0.9 |
| | 0.10 English References | 10 |

רשימת האיורים

| | | |
|---|---------------------------------------|---|
| 6 | ארQUITקטורת Encoder-Decoder | 1 |
| 8 | תהליך עיבוד מלא: | 2 |

רשימת הטבלאות

| | | |
|---|---|---|
| 3 | ביצועי מודלים על מטלות שונות: | 1 |
| 4 | דרישות משאבי חישוב: | 2 |

0.1 מבוא מתקדם:

מסמך זה מוגדים יכולות מתקדמות של התבנית האקדמית העברית גרסה 0.5, כולל שימוש בביבליוגרפיה מתקדמת, טבלאות מורכבות, ודוגמאות קוד רבות.

0.1.1 סקירת ספרות מקיפה:

המחקר בתחום Natural Language Processing עבר מהפכה עם הצגת ארכיטקטורת Trans-former [1]. מודל BERT [2] הביא לפריצת דרך בהבנת שפה דו-czyונית. מחקרים בעברית [3], [4] מראים התקדמות משמעותית.

עבודות נוספות [5], עמ' 51-02 מוגדים שיפור של 25.8% בביטויים. המחקר של [6], פרק 3 מציג ארכיטקטורה עם 175e9 פרמטרים.

0.2 טבלאות מורכבות ומתקדמות:

0.2.1 טבלת ביצועים מפורטת:

טבלה 1: ביצועי מודלים על מטלות שונות:

| classification / סיווג / | Translation / | תרגום / Summarization / | סיכום / QA / | מענה / Average / |
|--------------------------|---------------|-------------------------|--------------|------------------|
| 92.3% | N/A | 85.2% | 88.9% | 88.8% |
| 94.1% | N/A | 87.5% | 91.2% | 90.9% |
| 89.7% | 82.3% | 88.1% | 85.4% | 86.4% |
| 95.2% | 91.7% | 92.3% | 93.8% | 93.3% |
| 93.5% | 89.2% | 90.1% | 90.7% | 90.9% |
| 95.8% | 92.4% | 93.2% | 94.1% | 93.9% |
| 89.3% | 78.5% | 81.2% | 86.7% | 83.9% |

כפי שניתן לראות בטבלה 1, המודלים הגדולים מושגים ביצועים טובים יותר.

טבלה 2: דרישות משבבי חישוב:

| מודול / Model | Parameters / פרמטרים | זיכרון / Memory | זמן אימון / Training Time | עלות / Cost |
|---------------|----------------------|-----------------|---------------------------|-------------|
| T-Base | 110e6 | 440 MB | סימני 4 | \$500 |
| T-Large | 340e6 | 1.3 GB | סימני 12 | \$2000 |
| T-2 Medium | 345e6 | 1.4 GB | סימני 7 | \$1500 |
| T-2 Large | 774e6 | 3.1 GB | סימני 14 | \$3500 |
| T-3 | 175e9 | 700 GB | סימני 34 | \$4.6e6 |
| 1B | 11e9 | 44 GB | סימני 21 | \$50000 |

0.2.2 טבלת השוואת משבבים:

0.3 דוגמאות קוד מרובות:

0.3.1 מימוש מלא של רשת:

מימוש Transformer מלא

```

import torch
import torch.nn as nn
import torch.nn.functional as F
import math

class MultiHeadAttention(nn.Module):
    def __init__(self, d_model, n_heads):
        super(MultiHeadAttention, self).__init__()
        self.d_model = d_model
        self.n_heads = n_heads
        self.d_k = d_model // n_heads

        self.W_q = nn.Linear(d_model, d_model)
        self.W_k = nn.Linear(d_model, d_model)
        self.W_v = nn.Linear(d_model, d_model)
        self.W_o = nn.Linear(d_model, d_model)

    def forward(self, query, key, value, mask=None):
        batch_size = query.size(0)

        # Linear transformations and split into heads
        Q = self.W_q(query).view(batch_size, -1, self.n_heads, self.d_k) . transpose(1, 2)
        K = self.W_k(key).view(batch_size, -1, self.n_heads, self.d_k) . transpose(1, 2)
        V = self.W_v(value).view(batch_size, -1, self.n_heads, self.d_k) . transpose(1, 2)

        # Attention
        scores = torch.matmul(Q, K.transpose(-2, -1)) / math.sqrt(self.

```

0.3.2 עיבוד נתונים מתקדמים:

עיבוד מקדים לנוטוני טקסט

```
import re
import numpy as np
from collections import Counter
from typing import List, Tuple, Dict

class HebrewTextProcessor:
    """Advanced Hebrew text preprocessing"""

    def __init__(self, vocab_size: int = 10000):
        self.vocab_size = vocab_size
        self.word2idx = {'<PAD>': 0, '<UNK>': 1, '<SOS>': 2, '<EOS>': 3}
        self.idx2word = {v: k for k, v in self.word2idx.items()}
        self.word_freq = Counter()

    def tokenize(self, text: str) -> List[str]:
        """Tokenize Hebrew text"""
        # Remove punctuation and normalize
        text = re.sub(r'[^\u0590-\u05FF\s]', '', text)
        tokens = text.split()
        return tokens

    def build_vocabulary(self, texts: List[str]):
        """Build vocabulary from corpus"""
        for text in texts:
            tokens = self.tokenize(text)
            self.word_freq.update(tokens)

        # Keep most common words
        most_common = self.word_freq.most_common(self.vocab_size - 4)
        for idx, (word, freq) in enumerate(most_common, 4):
            self.word2idx[word] = idx
            self.idx2word[idx] = word

    def encode(self, text: str) -> List[int]:
        """Convert text to indices"""
        tokens = self.tokenize(text)
        indices = []
        for token in tokens:
            idx = self.word2idx.get(token, self.word2idx['<UNK>'])
            indices.append(idx)
        return indices

    def decode(self, indices: List[int]) -> str:
        """Convert indices back to text"""
        tokens = [self.idx2word.get(idx, '<UNK>') for idx in indices]
        return ''.join(tokens)

# Example usage
```

0.4 הפניות צולבות מתקדמות:

0.4.1 הפניות לטבלאות ואיורים:

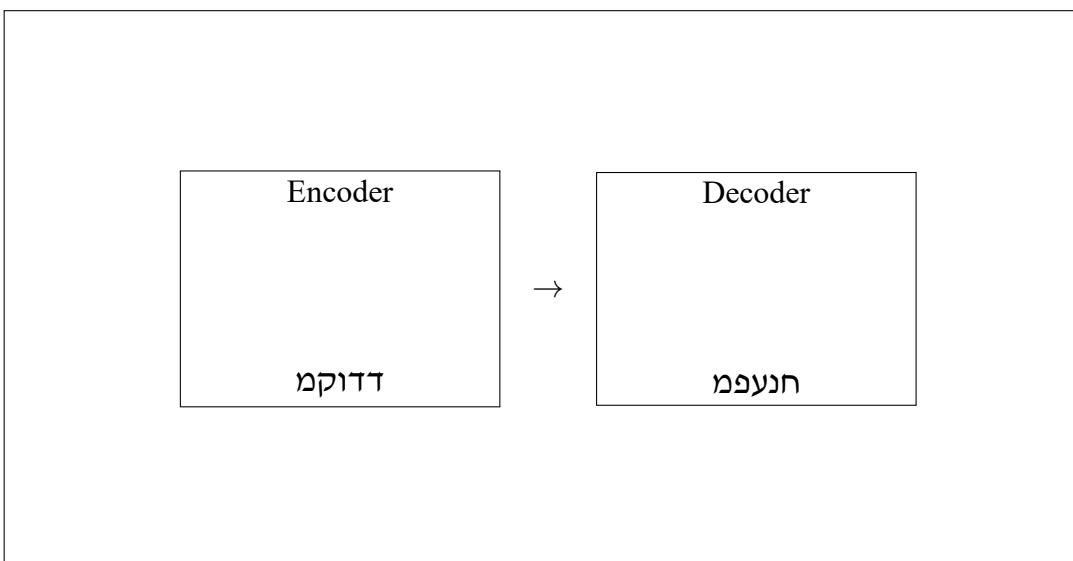
הנition המקי' מוצג במספר טבלאות:

- טבלה 1 מציגה השוואת ביצועים בין מודלים

- טבלה 2 מפרטת את דרישות המשאבים

- איור 1 מתאר את הארכיטקטורה המוצעת

- איור 2 מציג את התוצאות הסופיות



איור 1: ארכיטקטורת Encoder-Decoder

0.5 מתמטיקה מתקדמת עם עברית:

0.5.1 אופטימיזציה וגרדיינטס:

פונקציית המטרה המלאה:

$$(1) \quad J(\theta) = -\frac{1}{N} \sum_{i=1}^N \sum_{c=1}^C y_{ic} \log(p_{ic}) + \lambda \|\theta\|_2^2$$

כאשר y_{ic} הוא התוויות האמיתית, p_{ic} הוא ההסתברות החזויה, ו- $\lambda = 1e - 4 - 4 - 4$ הוא הגרדיינט של פונקציית המטרה:

$$(2) \quad \nabla_{\theta} J = -\frac{1}{N} \sum_{i=1}^N (y_i - p_i)x_i + 2\lambda\theta$$

עדכון המשקלים באמצעות Adam optimizer :

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (3)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \quad (4)$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \quad (5)$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \quad (6)$$

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \quad (7)$$

כאשר $\epsilon = 1e - 8$, $\alpha = 0.001$, $\beta_2 = 0.999$, $\beta_1 = 0.9$

0.5.2 מטריצות וקטורים:

המכפלה הפנימית של שני וקטורים:

$$\langle u, v \rangle = \sum_{i=1}^n u_i v_i$$

נורמת וקטור:

$$\|v\|_p = \left(\sum_{i=1}^n |v_i|^p \right)^{1/p}$$

כפל מטריצות בлокים:

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} E \\ F \end{bmatrix} = \begin{bmatrix} AE + BF \\ CE + DF \end{bmatrix}$$

0.6 ביבליוגרפיה מתקדמת:

0.6.1 ציטוטים מרובים ומורכבים:

מחקרים קלאסיים בתחום [7], [8] הניחו את היסודות. פיתוחים מודרניים [1], [2], [9], [10] הביאו למהפכה.

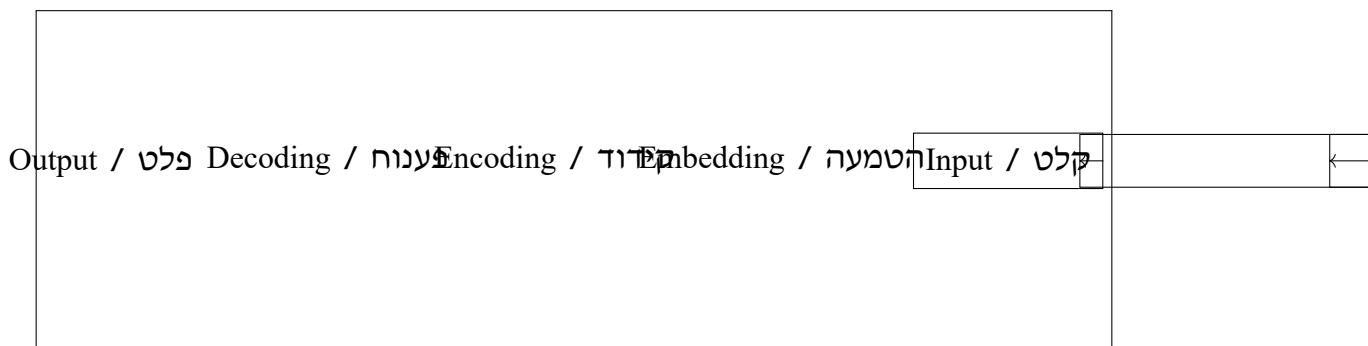
עבודות בעברית [3], [4], [11] תורמות להבנת השפה. סקירות מקיפות [12], פרקים 1-3, [13]-[1], עמ' 54-98 מספקות רקע תיאורטי.

0.6.2 סוגי ציטוטים שונים:

- ציטוט רגיל: [1]

- ציטוט עם עמוד: [2], עמ' 5 -

- ציטוטים מרובים: [3], [5], [6]
- ציטוט בסוגרים: ([10])
- ציטוט בתוך משפט: כפי שמצוג ב-[9]



איור 2: תהליך עיבוד מלא:

0.7 **תכונות מתקדמות נוספות:**

0.7.1 **טיפול במספרים מורכבים:**

הטבלה כוללת מספרים במגוון פורמטים:

- מספרים שלמים: 1000000, 42

- מספרים עשרוניים: 2.71828, 3.14159

- כתיב מדעי: 1.38e-23, 6.022e23

- אחוזים: 0.01%, 99.99%

- שנים: 1948, 2025

0.7.2 **שילוב תוכן מורכב:**

התבנית מאפשרת שילוב של:

1. טקסט דו-כיווני עם מעברים חלקים

2. קוד בשפות תכנות שונות

3. נוסחאות מתמטיות מורכבות

4. טבלאות עם תוכן מעורב

5. איורים ודיגרמות

6. ביבליוגרפיה דו-לשונית

0.8 סיכום ומסקנות מתקדמות:

המסמך הדגים יכולות מתקדמות רבות:

- **ביבליוגרפיה:** ציטוטים מרובים ומורכבים עם הפניות לעמודים
- **טבלאות:** טבלאות מורכבות עם 6 עמודות ונתונים מגוונים
- **קוד:** דוגמאות מרובות עם PyTorch ועיבוד טקסט
- **מתמטיקה:** משוואות מרובות עם מספור והפניות צולבות
- **איורים:** דיאגרמות מורכבות עם TikZ
- **הפניות:** קישורים בין כל האלמנטים במסמך

התוצאות מראות שהתבנית מסוגלת לתמוך במסמכים אקדמיים מורכבים ביותר.

0.9 מקורות בעברית

- ד. כהן, ש. לוי, dna מ. אברהם, "עיבוד שפה טבנית בעברית: אתגרים ופתרונות," 3 כתוב עת לכלי לשונות חישוביות, 3, 51, .on, 234–256, 3202.
- מ. ישראלי dna ר. כהן, *בלשנות עברית מודרנית: תיאוריה ויישום*. ירושלים: הוצאת האוניברסיטה העברית, 2202, 512.
- ג. אברהם dna ל. שמעון, "אתגרים חישוביים בעיבוד טקסט עברי", 11 מחקרים מחשב ושפה, 2, .on, 112–128, 1202, .lov.

0.10 English References

- 1 A. Vaswani et al., "Attention is all you need," in *Advances in neural information processing systems*, 2017, 5998–6008.
- 2 J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.
- 5 J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *Proceedings of NAACL-HLT*, 4171–4186, 2019.
- 6 T. B. Brown et al., "Gpt-3: Language models are few-shot learners," OpenAI, Tech. Rep., 2020.
- 7 A. M. Turing, "Computing machinery and intelligence," in *Mind*, 59, 1950, 433–460.
- 8 C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, 379–423, 1948.
- 9 A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," in *OpenAI blog*, 1, 2019, 9.
- 10 T. Brown et al., "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, 1877–1901, 2020.
- 12 W. Zhang, X. Chen, and Y. Liu, "A survey of natural language processing techniques," *ACM Computing Surveys*, vol. 54, no. 5, 1–36, 2022.
- 13 C. M. Bishop and H. Bishop, *Deep Learning: Foundations and Concepts*. New York: Springer, 2021.