

# סיכום שיעור

## חלונות הקשר ו-RAG

### Context Windows and RAG

ד"ר יורם סגל

כל הזכויות שמורות לד"ר יורם סגל (C)

## מילות מפתח

-- חלון הקשר -- Context Window  
-- יצירה מוגברת באחזור -- RAG -- Retrieval-Augmented Generation  
-- מודלי שפה גדולים -- LLM -- Large Language Models  
-- טוקנים -- Tokens  
-- וקטורים -- Vectors  
-- צ'אנקים -- Chunks  
-- חיפוש סמנטי -- Semantic Search  
-- מסד נתונים וקטורי -- Vector Database  
-- עקומת ה-U -- Lost in the Middle  
-- הנחיית מערכת -- System Prompt

## 1 חלון ההקשר -- המודל והאנלוגיה

### 1.1 הבנת המודל הבסיסי

מודלי שפה גדולים (LLMs) כמו GPT ו-Claude הם חסרי מצב (stateless) -- כל שיחה מתחילה מאפס ואין להם זיכרון משיחות קודמות. הפתרון הוא חלון ההקשר (Context Window), שמאפשר לנהל את המידע בכל שיחה.

### 2.1 האנלוגיה למחשב

ניתן להבין את המערכת באמצעות אנלוגיה למחשב:

-- ה-LLM שקול ל-CPU -- המעבד שמבצע את החישובים  
-- חלון ההקשר שקול ל-RAM -- הזיכרון הזמני  
-- מסד הנתונים הוקטורי שקול ל-Hard Disk -- אחסון קבוע  
-- המשתמש הוא מערכת ההפעלה -- מנהל את זרימת המידע

חשוב להבין את היררכיית המהירות: רגיסטרים הם המהירים ביותר (ויקרים ביותר), אחריהם RAM, Cache, ולבסוף Hard Disk (איטי אך זול). ההבדל במהירות בין RAM ל-Hard Disk הוא פי אלף -- כמו ההבדל בין טיסה להודו לבין הליכה ברגל.

### 3.1 טוקנים ווקטורים

כל טקסט מומר לטוקנים, וכל טוקן מיוצג כווקטור. כללי אצבע חשובים:

- באנגלית: כל טוקן מכסה כ-0.7 מילה
- בעברית: כל מילה צורכת כ-2.5 טוקנים (בגלל שימוש נמוך יחסית בעברית באימון)
- הווקטור מורכב מקומפוננטות שכל אחת מייצגת מאפיין סמנטי

### 4.1 עקומת ה-U -- אובדן מידע באמצע

מחקר פורץ דרך מ-2023 גילה תופעה קריטית: מודלי שפה שוכחים מידע שנמצא באמצע חלון ההקשר. הביצועים במציאת מידע:

- בהתחלה -- גבוהים מאוד (Attention חזק)
  - באמצע -- צונחים ל-50%
  - בסוף -- חוזרים לרמה גבוהה
- המסקנה המעשית:** אל תשימו מידע קריטי באמצע! מידע חשוב צריך להופיע בתחילת החלון ההקשר או בסופו.

### 5.1 אסטרטגיות לניהול חלון ההקשר

ארבע אסטרטגיות מרכזיות:

1. **כתיבה תמציתית** (Write) -- שימוש בתבניות ומזעור טקסט מיותר, קבצים קטנים (עד 150 שורות)
2. **סיכום תקופתי** (Select) -- סיכום חלון ההקשר לאחר כל שאילתה
3. **אחזור סלקטיבי** (Compress) -- שימוש בחיפוש סמנטי לאחזור המסמכים הרלוונטיים ביותר
4. **בידוד** (Isolate) -- שמירה על חלונות הקשר נפרדים לסוכנים או משימות שונות

### 6.1 הכלכלה של חלון ההקשר

יש הבדלים משמעותיים במחיר בין מודלים:

- Gemini 1.5 Flash -- 128,000 טוקנים, \$0.15 למיליון (הבסיס)
  - Claude 3.5 Sonnet -- 200,000 טוקנים, \$3.00 למיליון (פי 20 יקר יותר)
  - GPT-4o -- 128,000 טוקנים, \$2.50 למיליון
  - Gemini 1.5 Pro -- 2,000,000 טוקנים, \$1.25 למיליון
- מפתחים ומנהלים חייבים לשקול את האיזון בין ביצועים, גודל חלון ההקשר ועלויות.

## 2 RAG -- יצירה מוגברת באחזור

### 1.2 הבעיה היסודית

מודלי שפה גדולים הם כמו גאונים עם זיכרון מושלם, אך עם מגבלות:

- סטטיים -- הידע שלהם קפוא בזמן האימון
- כלליים -- אין להם גישה למידע ספציפי של הארגון
- מוגבלים -- לא יכולים לקרוא מיליוני מסמכים בבת אחת

### 2.2 הפתרון -- הספרן הדיגיטלי

RAG (Retrieval-Augmented Generation) היא טכניקה שהופכת את המודל לחכם יותר:

1. שאלתה -- המשתמש שואל שאלה
2. חיפוש -- המערכת מחפשת במאגר מסמכים חיצוני
3. אחזור -- שליפת המסמכים הרלוונטיים ביותר (Top-K)
4. בניית הקשר -- הוספת המסמכים לחלון ההקשר
5. יצירת תשובה -- המודל עונה על בסיס המידע שהוחזר

### 3.2 ארכיטקטורת RAG

המערכת מורכבת מארבעה רכיבים:

- Query Encoder -- מקודד את השאלתה לוווקטור
  - Document Encoder -- מקודד מסמכים לוווקטורים (חד פעמי)
  - מנוע אחזור -- מחשב מרחק וקטורי (בדרך כלל קוסינוס)
  - Generator -- מייצר תשובה על בסיס המסמכים
- חשוב: Query Encoder ו-Document Encoder חייבים להשתמש באותו מודל!

### 4.2 שתי גישות ל-RAG

1. Sequence RAG -- אחזור בתחילת התהליך, יצירת תשובה שלמה
2. Dynamic RAG -- אחזור מרובד במהלך יצירת התשובה (כמו Deep Research)

### 5.2 צ'אנקינג -- פיצול מסמכים

שלוש שיטות עיקריות:

- לפי טוקנים -- כל 512 טוקנים עם חפיפה של 50. פשוט אך עלול לחתוך באמצע רעיון
- לפי משפטים -- צירוף משפטים עד 400 טוקנים. שומר על שלמות
- סמנטי -- פיצול לפי נושאים עם Embedding. מורכב אך מדויק

## 6.2 מסדי נתונים וקטוריים

המהפכה של 2019 (Sentence-BERT) אפשרה ליצור Embedding עצמאי לכל מסמך ולשמור אותו. מסדי נתונים וקטוריים כמו Chroma מאפשרים חיפוש מהיר פי אלף.

## 7.2 דוגמאות מעשיות

- GPTs -- יצירת סוכן עם בסיס ידע ב-OpenAI
- Claude Projects -- הוספת מסמכים לפרויקט ב-Claude
- Perplexity -- מנוע חיפוש שמסכם אתרים

## 8.2 בעיות נפוצות ב-RAG

- אין מידע -- המידע לא קיים במאגר
  - Not in Top-K -- המסמך הרלוונטי לא בתוצאות הראשונות
  - Lost in the Middle -- המידע באמצע ולא נמצא
  - Not Extracted -- המודל לא חילץ נכון (בעיקר בטבלאות)
  - שרשרת כישלון -- כל שלב תלוי בקודמו
- פתרון מומלץ: השתמשו ב-Few-Shot Examples! תנו דוגמאות של קלט ופלט רצויים.

## 3 הזכות לשכוח -- היבט פילוסופי

### 1.3 חלון ההקשר המוגבל כפיצ'ר

- חלון ההקשר המוגבל אינו באג -- הוא פיצ'ר. השכחה מאפשרת:
- שחרור מהעבר -- סוכן שטעה לא סוחב את הטעות אלפי צעדים קדימה
  - גמישות ויעילות -- לא כל מידע חשוב לנצח
  - התחלה מחדש -- כמו הזכות לשכוח במשפט נגד גוגל

### 2.3 מה מבדיל אותנו מ-AI?

- רגש אמיתי -- לסוכן יש רק סטטיסטיקה של רגש
- הזכות לטעות בכוונה -- אנחנו יכולים לבחור ללכת נגד הווייז
- תשומת לב מוגבלת -- אנחנו בוחרים במה להתמקד
- חוויה -- אנחנו נהנים וסובלים, לא רק מעבדים

### 3.3 כלכלת AI חיונית ושליטת

תמיד חייב להיות אדם בלולאה (Human-in-the-Loop) שיכול:

- להוריד את המתג
- לקבוע סדר עדיפויות
- לשמור על מיקוד ולא להיסחף

## 4 המטלה -- ניסוי מחקרי

### 1.4 גישה מחקרית

המטלה הפעם שונה -- לא תכנות אלא מחקר:

- נסחו שאלת מחקר
- הציבו השערה
- תכננו ניסוי
- הציגו ממצאים בצורה יצירתית

### 2.4 נושאי מחקר מוצעים

1. **Lost in the Middle** -- בדיקת ירידה בדיוק כשמידע באמצע
2. **השפעת גודל** -- כיצד גודל חלון ההקשר משפיע על ביצועים
3. **אסטרטגיות ניהול** -- השוואת Write/Select/Compress/Isolate
4. **הצטברות בעיות** -- כיצד בעיות מצטברות בסוכנים