

Jegyzőkönyv

Webtechnológia
Féléves feladat

LSTM
Bemutató weboldal

Készítette: **Ferencsik Róbert**

Neptunkód: **BQLOTW**

Dátum: **2025. November**

Miskolc, 2025

Tartalomjegyzék

Table of Contents

Tartalomjegyzék.....	2
1. A Projekt Célja.....	3
2. Felhasznált Technológiák.....	3
3. Projektstruktúra.....	4
4. Funkcionalitás.....	4
Kódrészletek.....	5
HTML egyéni videó vezérlés.....	5
CSS Dinamikus tartalom stílusának megadására.....	6
JavaScript űrlap mező validáció.....	7
DOM-manipuláció dinamikus tartalom betöltése AJAX-szal és animációkkal.....	8

1. A Projekt Célja

A projekt célja egy informatív és felhasználóbarát weboldal létrehozása, amely bemutatja a Hosszú-Rövid Távú Memóriával (Long Short-Term Memory, LSTM) rendelkező neurális hálózatokat. Az oldal célközönsége a gépi tanulás iránt érdeklődők, nem pedig akademikusok.

Az oldal a következő főbb témaköröket öleli fel:

Kezdőlap: Általános, rövid bevezető az LSTM hálózatokba.

Történet: Az LSTM és a rekurrens hálózatok fejlődésének történeti áttekintése.

Architektúra: Az LSTM cellák és kapuk (gate-ek) részletes bemutatása.

Példa: Egy gyakorlati példa, vizualizáció az LSTM működéséről.

Kapcsolat: Egy űrlap, ahol a felhasználók kérdéseket tehetnek fel.

2. Felhasznált Technológiák

A weboldal a következő alapvető webtechnológiákra épül:

HTML5: A weboldalak szemantikus struktúrájának kialakításáért felel.

CSS3: A vizuális megjelenésért, a dizájnért és a reszponzív kialakításért felel. A stíluslapok külön fájlokban, logikailag szervezve találhatók (global.css, header.css, forms.css stb.).

JavaScript: A kliensoldali dinamikus funkcionalitásért felel.

jQuery: A DOM-manipuláció és eseménykezelés megkönnyítésére használt JavaScript könyvtár.

3. Projektstruktúra

A projekt fájlrendszere logikusan van felépítve, a tartalom, a stílus és a funkcionalitás szétválasztásának elve mentén.

Gyökérkönyvtár: Itt találhatóak a fő HTML oldalak (index.html, history.html, contact.html stb.).

src/ mappa: A projekt forráskódjának központi helye.

styles/: A CSS stíluslapokat tartalmazza, külön fájlokba bontva az egyes oldalak és komponensek szerint.

script/: A JavaScript fájlokat tartalmazza, amelyek a weboldal interaktív működését biztosítják.

img/: A weboldalon használt képeket és vizuális anyagokat tárolja.

data/: Adatfájlokat (pl. data.json) tartalmaz, amelyeket a szkriptek dinamikusan betölthetnek.

4. Funkcionalitás

A weboldal több dinamikus funkciót is tartalmaz a felhasználói élmény javítása érdekében:

animations.js: Vizuális animációkat kezel, valószínűleg a felhasználói interakciók során vagy elemek megjelenésekor.

form-validation.js: A kapcsolati űrlap kliensoldali validációját végzi. Ellenőrzi a kötelező mezőket (név, email, üzenet stb.) és a formátumok helyességét, majd visszajelzést ad a felhasználónak.

data-loader.js: Felelős a data.json fájl aszinkron betöltéséért és az adatok feldolgozásáért, valószínűleg a "Példa" oldalon.

dom-manipulation.js: Általános DOM-manipulációs segédfüggvényeket tartalmazhat.

video-controls.js: Egyedi vezérlőket implementál a beágyazott videókhöz (how-lstm-works.mp4).

script.js: A központi szkript, amely összefogja és inicializálja a többi szkript működését.

5. Kódrészletek

HTML egyéni videó vezérlés

Ez az `example.html` fájlból származó kódrészlet egy egyéni videolejátszó HTML-szerkezetét mutatja be. A böngésző alapértelmezett vezérlőelemei helyett egyéni gombok és legördülő menü van definiálva. Ez lehetővé teszi a lejátszó megjelenésének teljes körű vezérlését, valamint egyéni funkciók hozzáadását JavaScript segítségével.

```
<div class="video-container">
  <h2>Understanding LSTM Processing</h2>
  <video id="lstm-video" width="100%" controls>
    <source src="src/img/how-lstm-works.mp4" type="video/mp4">
    Your browser does not support the video tag.
  </video>
  <div class="video-controls">
    <button id="playBtn" class="video-control-btn">> Play</button>
    <button id="pauseBtn" class="video-control-btn">⏸ Pause</button>
    <button id="muteBtn" class="video-control-btn">🔊 Mute</button>
    <div class="speed-control">
      <label for="speedSelect">Speed:</label>
      <select id="speedSelect">
        <option value="0.5">0.5x</option>
        <option value="1" selected>1x</option>
        <option value="1.5">1.5x</option>
        <option value="2">2x</option>
      </select>
    </div>
    <button id="fullscreenBtn" class="video-control-btn">⏏ Fullscreen</button>
  </div>
</div>
```



CSS Dinamikus tartalom stílusának megadására

A `src/styles/example.css` fájlban található `text-prediction-demo` és `sequence-step` osztályok CSS-je tiszta és modern megjelenést biztosít a dinamikus generált tartalomnak. A bal oldali szegély használata egy jellegzetes színnel (`#4a90e2`) segít vizuálisan csoportosítani a kapcsolódó tartalmakat és felhívni a felhasználó figyelmét. Ez egy egyszerű, de hatékony módszer a felhasználói élmény javítására.

```
.text-prediction-demo {  
    background: #f8f9fa;  
    padding: 20px;  
    border-radius: 8px;  
    border-left: 4px solid #4a90e2;  
}
```

```
.sequence-step {  
    margin-bottom: 15px;  
    padding: 10px;  
    background: white;  
    border-radius: 5px;  
}
```

```
.step-label {  
    font-weight: bold;  
    color: #4a90e2;  
}
```

Example: Text Prediction

When predicting the next word in a sentence, an LSTM processes each word sequentially:

Input: "The cat sat on the"

LSTM processes: Each word maintains context from previous words

Prediction: "mat" (high probability based on learned patterns)

JavaScript űrlap mező validáció

Ez a javascript kód részlet regex használatával ellenőrzi a beviteli mezőbe írt emailcím helyességét. Az email címet tartalmazó elemet JQuery segítségével találja meg, majd a regex egy alap email struktúrára szűr, (karakterek)@(karakterek).(karakterek). Majd a feltételben eltávolítja a whitespace karaktereket az input elejéről és végéről. Ha így üres stringet kap (nincs bemenet), vagy nem illeszkedik a minta, akkor egy hibaüzenetet kapunk.

```
const email = $('#email');
const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
if (!email.val().trim() || !emailRegex.test(email.val())) {
    showError(email, 'email-error', 'Please enter a valid email address');
    isValid = false;
}
```

Name *

Robert

Email *

robert@rob

Please enter a valid email address

DOM-manipuláció dinamikus tartalom betöltése AJAX-szal és animációkkal

A `src/script/data-loader.js` fájlban található kód több fejlett technikát is bemutat. A jQuery `\$.ajax` funkcióját használja az adatok aszinkron letöltésére egy JSON fájlból. Sikeres letöltés után dinamikusan generál HTML-tartalmat a szekvencia vizualizálásához. Ezután a jQuery animációs funkcióit használja az új tartalom fokozatos megjelenítéséhez és az egyes elemek egymást követő animálásához, így sima és vonzó felhasználói élményt teremtve. Ez egy hatékony kombináció dinamikus és interaktív weboldalak létrehozásához.

```
function loadSequenceData() {
    const container = $('#sequence-container');

    if (container.length === 0) return;

    $.ajax({
        url: 'src/data/data.json',
        method: 'GET',
        dataType: 'json',
        success: function (data) {
            displaySequenceData(data);
        },
        error: function (xhr, status, error) {
            console.error('Error loading JSON data:', error);
            container.html('<p>Error loading sequence data. Please try again
later.</p>');
        }
    });
}

function displaySequenceData(data) {
    const container = $('#sequence-container');
    let html = '';

    // ... (HTML generation logic) ...

    container.html(html);
}
```



```

// Enhanced animations
container.find('.sequence-card-enhanced').each(function (index) {
  const $card = $(this);
  $card.css({
    opacity: 0,
    transform: 'translateY(30px) scale(0.95)'
  }).delay(index * 300).animate({
    opacity: 1
  }, 800, function () {
    $card.css('transform', 'translateY(0) scale(1)');

    // ... (further animations for inner elements) ...
  });
});
}

```

