

Формализация информации и Big Data

02.03.03 -Математическое обеспечение и администрирование информационных систем,
направленность (профиль) -разработка и администрирование информационных систем

<http://vikchas.ru>

Лекция 7. Тема «Гетерогенный ИИ - формализация информации, Биг Дата и современные технологии»

Часовских Виктор Петрович

доктор технических наук, профессор кафедры
ШИиКМ, ФГБОУ ВО «Уральский государственный
экономический университет

Екатеринбург 2026

Понятие гетерогенного ИИ

Гетерогенный ИИ представляет собой подход к искусственному интеллекту, в котором сочетаются различные типы моделей, архитектур и вычислительных ресурсов для решения сложных задач. В контексте WEB-технологий это означает интеграцию разнородных ИИ-систем в единую экосистему WEB-приложений и сервисов.

Ключевые компоненты гетерогенного ИИ в WEB-среде

1. Многомодальные ИИ-модели

Тип модели	Применение в WEBе	Технологии реализации
Текстовые модели(LLM)	Чатботы, генерация контента, SEO-оптимизация	GPT-4, LLaMA, Claude, Transformers.js
Визуальные модели	Генерация и обработка изображений, компьютерное зрение	DALL-E, Stable Diffusion, TensorFlow.js
Аудиомодели	Распознавание речи, генерация аудио	Whisper API, Bark, Web Speech API
Мультимодальные модели	Комплексная обработка разных типов данных	GPT-4V, Gemini, Multimodal Transformers

2. Архитектурные подходы интеграции ИИ в WEB

Клиентские решения

- **Браузерный ИИ**

- Запуск легких моделей непосредственно в браузере с использованием WebGL и WebAssembly
- TensorFlow.js, ONNX.js для выполнения моделей на клиентской стороне
- Снижение нагрузки на сервер и обеспечение конфиденциальности

Серверные решения

- **API-ориентированная архитектура**

- Доступ к мощным моделям через REST/GraphQL API
- Микросервисы для различных ИИ-функций
- Оркестрация нескольких ИИ-сервисов

Гибридные подходы

- Комбинирование локальной предобработки с облачными инференсами
- Адаптивное переключение между легкими локальными и мощными удаленными моделями
- Progressive enhancement с ИИ-функциональностью

Интеграция гетерогенного ИИ в современные WEB-технологии

1. Фронтенд-интеграции

Технология	Методы интеграции ИИ	Примеры применения
React.js	- Хуки для ИИ-функциональности - Компоненты с ИИ-возможностями - React Server Components для серверного ИИ	Интеллектуальные формы, предиктивный UI, адаптивный дизайн
Vue.js	- Композиционное API для ИИ-сервисов - Плагины для интеграции моделей	Чат-интерфейсы, умные фильтры контента
Angular	- Сервисы для ИИ-обработки - Директивы для ИИ-компонентов	Бизнес-аналитика, умные дашборды
WebComponents	- Изолированные ИИ-компоненты - Shadow DOM для ИИ-виджетов	Переиспользуемые ИИ-элементы между фреймворками

2. Бэкенд-решения для гетерогенного ИИ

Технология	Возможности для ИИ	Особенности
Node.js	- Асинхронная обработка ИИ-запросов - Легкая интеграция с JS-моделями - Стриминг ИИ-результатов	Хорошо для оркестрации ИИ-сервисов и API-шлюзов
Python (FastAPI/Flask)	- Нативная интеграция с PyTorch, TensorFlow - Богатая экосистема ИИ-библиотек - Асинхронная обработка	Оптимально для сервисов инференса и обучения моделей
ASP.NETCore	- ML.NET интеграция - Высокопроизводительные ИИ-эндпоинты - gRPC для ИИ-коммуникации	Корпоративные решения с ИИ-компонентами
Go	- Высокопроизводительные ИИ-прокси - Параллельная обработка ИИ-запросов	Сервисы с высокой нагрузкой и низкой латентностью

3. Облачные решения и сервисы

Поставщик	Сервисы для гетерогенного ИИ	Интеграция с WEB
AWS	- Bedrock для мультимодальных моделей - SageMaker для ML-операций - Lambda для serverless ИИ	API Gateway, AppSync для ИИ-интеграций
Azure	- Azure OpenAI Service - Cognitive Services - Custom Vision	Интеграция с Azure Static Web Apps, App Service

Поставщик	Сервисы для гетерогенного ИИ	Интеграция с WEB
Google Cloud	- Vertex AI Hub - AI API - Gemini API	Cloud Functions, Cloud Run для ИИ-компонентов
Hugging Face	- Inference API - Spaces - Model Hub	Webhook-интеграции, SDK для JavaScript

Современные WEB-архитектуры с гетерогенным ИИ

1. JAMstack с ИИ-расширениями

- Статическая генерация с ИИ-оптимизацией контента
- Функции API для вызова разных ИИ-моделей
- Использование Edge Computing для низколатентного ИИ
- Примеры технологий: Next.js с ISR, Netlify Functions, Vercel Edge Functions

2. Микрофронтенды с ИИ-компонентами

- Отдельные команды разрабатывают специализированные ИИ-микрофронтенды
- Федерация модулей с общими ИИ-возможностями
- Изоляция тяжелых ИИ-компонентов
- Примеры: Module Federation, Web Components, single-spa

3. Архитектура на основе потоков данных для ИИ

- Реактивный подход к обработке данных для ИИ
- Event-driven архитектура для интеграции разнородных ИИ
- Real-time обновления на основе ИИ-инсайтов
- Технологии: RxJS, Redux Observable, Akka.js

Примеры реализации гетерогенного ИИ в WEB-приложениях

1. Интеллектуальный редактор контента

JAVASCRIPT

// React-компонент с использованием нескольких ИИ-моделей

```
function SmartEditor() {  
  const [content, setContent] = useState("");  
  const [suggestions, setSuggestions] = useState([]);  
  
  // Текстовая модель для автозаполнения  
  const getTextSuggestions = useCallback(debounce(async (text) => {  
    const response = await textCompletionAPI.complete(text);  
    setSuggestions(response.suggestions);  
  }, 300), []);  
  
  // Визуальная модель для генерации изображений  
  const generateImage = async (prompt) => {  
    const imageData = await imageGenerationAPI.generate(prompt);  
    return imageData.url;  
  };  
  
  // Мультимодальная модель для семантического анализа  
  const analyzeContent = async () => {  
    const analysis = await multimodalAPI.analyze({  
      text: content,  
      images: document.querySelectorAll('img').src  
    });  
    return analysis;  
  };  
};
```

```
// ... остальной код компонента
```

```
}
```

2. WEB-приложение с гетерогенной ИИ-аналитикой

TYPESCRIPT

```
// Сервис-оркестратор для разных ИИ-моделей (TypeScript)
```

```
class HeterogeneousAIOrchestrator {
```

```
  private textAnalyzer: TextAnalysisModel;
```

```
  private imageRecognizer: VisionModel;
```

```
  private predictionEngine: ForecastModel;
```

```
  private sentimentAnalyzer: EmotionModel;
```

```
  constructor() {
```

```
    // Инициализация разных моделей
```

```
    this.textAnalyzer = process.env.NODE_ENV === 'production'
```

```
      ? new CloudTextModel()
```

```
      : new LightweightTextModel();
```

```
    this.imageRecognizer = new WebGLVisionModel();
```

```
    this.predictionEngine = new TensorFlowJSModel();
```

```
    this.sentimentAnalyzer = new SentimentAPIClient();
```

```
  }
```

```
  async analyzeUserData(userData: UserData): Promise<AnalysisResult> {
```

```
    // Параллельный запуск разных типов анализа
```

```
    const [textResults, imageInsights, predictions, sentiment] = await Promise.all([
```

```
      this.textAnalyzer.process(userData.textContent),
```

```
      userData.images ? this.imageRecognizer.recognize(userData.images) : null,
```

```
      this.predictionEngine.forecast(userData.behaviorHistory),
```

```
      this.sentimentAnalyzer.analyze(userData.messages)
```

```

    });

    // Агрегация результатов разных моделей
    return this.aggregateResults({ textResults, imageInsights, predictions, sentiment
});
}

private aggregateResults(results: PartialResults): AnalysisResult {
    // Логика объединения результатов от разных моделей
    // ...
}
}

```

Перспективы и тренды развития

1. **Федеративное обучение в браузере**- распределенное улучшение моделей без передачи данных на сервер
2. **ИИ-ускоренные WEB-компоненты**- стандартизированные элементы с встроенным ИИ
3. **Персонализированные ИИ-агенты**- WEB-приложения с долгосрочной памятью о пользователях
4. **Мультиагентные системы в WEB-сервисах**- взаимодействие разных ИИ-агентов для решения комплексных задач
5. **WebGPU для ИИ**- использование графических возможностей для ускорения инференса в браузере

Гетерогенный ИИ в сочетании с современными WEB-технологиями создает основу для следующего поколения интеллектуальных WEB-приложений, сочетающих мощь разнообразных моделей с гибкостью и доступностью WEB-платформы.

Гетерогенный ИИ и современные WEB-технологии в среде Visual Studio

Интеграция гетерогенного ИИ в WEB-разработку с использованием Visual Studio

Visual Studio предоставляет мощную среду для разработки WEB-приложений с интеграцией гетерогенных ИИ-технологий. Рассмотрим ключевые аспекты этой экосистемы.

Экосистема Visual Studio для работы с ИИ

Компонент	Описание	Применение в контексте гетерогенного ИИ
Visual Studio 2022	Флагманская IDE с полной поддержкой .NET	Основная среда разработки для ИИ-интегрированных WEB-приложений
Visual Studio Code	Легковесный кросс-платформенный редактор	Быстрая разработка и тестирование JavaScript/Python ИИ-компонентов
GitHub Copilot	ИИ-ассистент для написания кода	Ускорение разработки WEB-приложений с ИИ-компонентами
Azure AI Studio	Интегрированное решение для ИИ-разработки	Создание и управление облачными ИИ-сервисами для WEB-приложений

WEB-технологии и фреймворки в Visual Studio для гетерогенных ИИ-решений

.NET-стек для WEB-разработки с ИИ

[ASP.NET](#)Core и ИИ

CSHARP

// Пример интеграции OpenAI в ASP.NET Core контроллер

[ApiController]

[Route("api/[controller]")]

```

public class AIController : ControllerBase
{
    private readonly OpenAIClient _openAIClient;

    public AIController(OpenAIClient openAIClient)
    {
        _openAIClient = openAIClient;
    }

    [HttpPost("generate-text")]
    public async Task<IActionResult> GenerateText(TextGenerationRequest request)
    {
        var chatCompletionsOptions = new ChatCompletionsOptions
        {
            DeploymentName = "gpt-4",
            Messages = { new ChatMessage(ChatRole.User, request.Prompt) }
        };

        var response = await _openAIClient.GetChatCompletionsAsync(chatCompletionsOptions);
        return Ok(new { Text = response.Value.Choices[0].Message.Content });
    }
}

```

[ML.NET](#) в Visual Studio

- Интеграция [ML.NET](#) Model Builder через визуальный интерфейс Visual Studio
- Поддержка AutoML для быстрого прототипирования моделей
- Трансформация моделей для WEB-развертывания

CSHARP

// Пример использования ML.NET для классификации изображений в WEB-приложении

```
[HttpPost("classify")]
```

```
public IActionResult ClassifyImage([FromForm] IFormFile image)
```

```
{
```

```
    using var ms = new MemoryStream();
```

```
    image.CopyTo(ms);
```

```
    // Используем предобученную модель ML.NET
```

```
    var imageData = new ImageData { Image = ms.ToArray() };
```

```
    var prediction = _mlContext.Model.Predict(imageData);
```

```
    return Ok(new {
```

```
        Label = prediction.PredictedLabel,
```

```
        Confidence = prediction.Score.Max()
```

```
    });
```

```
}
```

Blazor для интерактивных ИИ-приложений

Blazor позволяет создавать интерактивные WEB-интерфейсы с гетерогенным ИИ на стороне клиента и сервера:

CSHARP

```
// Компонент Blazor с интеграцией ИИ для анализа текста
```

```
@page "/sentiment-analysis"
```

```
@using Microsoft.ML.OnnxRuntime
```

```
@inject IJSRuntime JSRuntime
```

```
<div class="container">
```

```
    <textarea @bind="userInput" class="form-control" rows="5"></textarea>
```

```
    <button @onclick="AnalyzeSentiment" class="btn btn-primary mt-3">Анализировать</button>
```

```

@if (sentiment != null)
{
    <div class="alert @(sentiment > 0.5 ? "alert-success" : "alert-danger") mt-3">
        Настроение: @(sentiment > 0.5 ? "Позитивное" : "Негативное")
        (@Math.Round(sentiment.Value * 100)%)
    </div>
}
</div>

```

```

@code {
    private string userInput = "";
    private float? sentiment;

    private async Task AnalyzeSentiment()
    {
        // Используем JS-интероп для вызова TensorFlow.js модели
        sentiment = await
        JSRuntime.InvokeAsync<float>("analyzeSentimentWithTFJS", userInput);
    }
}

```

JavaScript и TypeScript интеграции

Visual Studio предоставляет отличную поддержку JavaScript/TypeScript для фронтенд интеграции ИИ-моделей:

TYPESCRIPT

// TensorFlow.js интеграция в Visual Studio

```
import * as tf from '@tensorflow/tfjs';
```

```
import * as toxicity from '@tensorflow-models/toxicity';
```

```
export class ContentModerator {
```

```

private model: toxicity.ToxicityClassifier | null = null;

async initialize() {
    // Загружаем модель модерации контента
    this.model = await toxicity.load(0.7, ['identity_attack', 'insult', 'threat']);
    console.log('Модель TensorFlow.js загружена');
}

async analyzeText(text: string): Promise<any> {
    if (!this.model) {
        throw new Error('Модель не инициализирована');
    }

    const predictions = await this.model.classify(text);
    return predictions.map(p => ({
        category: p.label,
        isToxic: p.results[0].match,
        confidence: p.results[0].probabilities[1]
    }));
}
}

```

Инструменты для разработки гетерогенного ИИ в Visual Studio

Расширения Visual Studio для ИИ-разработки

Расширение	Функционал	Применение в WEB-разработке
AI Tools for Visual Studio	Интеграция с Azure AI, локальные модели	Создание WEB-API с ИИ-функциональностью
Python Tools for VS	Поддержка Python, Jupyter Notebooks	Разработка бэкенд-сервисов с PyTorch, TensorFlow

Расширение	Функционал	Применение в WEB-разработке
IntelliCode	ИИ-подсказки для кода	Продуктивная разработка WEB-сервисов
ONNX Runtime	Интеграция моделей ONNX	Оптимизированное выполнение ИИ-моделей в WEBе

Azure DevOps и ИИ CI/CD для WEB-проектов

- Автоматизация тестирования ИИ-компонентов
- Мониторинг производительности моделей
- Интеграция с Azure ML для автоматического переобучения
- Канареечные развертывания ИИ-функциональности

Архитектурные шаблоны для WEB-проектов с гетерогенным ИИ в Visual Studio

Многоуровневая архитектура с ИИ-компонентами

[Клиентский уровень]

- |-- Blazor WebAssembly с TensorFlow.js
- |-- JavaScript фронтенд с ONNX.js
- |-- Progressive Web App с локальными моделями

[API уровень]

- |-- ASP.NET Core API с роутингом к ИИ-сервисам
- |-- SignalR для realtime ИИ-предсказаний
- |-- gRPC для высокопроизводительного ИИ

[Сервисный уровень]

- |-- ML.NET сервисы для локальных моделей
- |-- Интеграция с Azure AI Services
- |-- Прокси к внешним ИИ API (OpenAI, Hugging Face)

[Уровень данных]

```
-- Azure Cosmos DB для хранения ИИ-результатов
-- Blob Storage для медиа-данных
-- Azure Cache for Redis для кэширования ИИ-ответов
```

Шаблон оркестрации гетерогенных ИИ-моделей

CSHARP

// Пример оркестрации различных ИИ-моделей в WEB-приложении

```
public class AIOrchestrator
{
    private readonly IEnumerable<IAIModelService> _modelServices;
    private readonly ILogger<AIOrchestrator> _logger;

    public AIOrchestrator(IEnumerable<IAIModelService> modelServices,
        ILogger<AIOrchestrator> logger)
    {
        _modelServices = modelServices;
        _logger = logger;
    }

    public async Task<AIResponse> ProcessRequest(AIRequest request)
    {
        // Выбор подходящих моделей на основе запроса
        var eligibleModels = _modelServices
            .Where(m => m.CanHandle(request))
            .OrderByDescending(m => m.Confidence(request));

        foreach (var model in eligibleModels)
        {
            try
            {

```

```

        var result = await model.ProcessAsync(request);
        if (result.IsSuccessful)
        {
            return result;
        }

        _logger.LogInformation("Модель {ModelName} не справилась,
        пробуем следующую", model.Name);
    }
    catch (Exception ex)
    {
        _logger.LogError(ex, "Ошибка в модели {ModelName}", model.Name);
    }
}

return AIResponse.Failed("Ни одна модель не смогла обработать запрос");
}
}

```

Практические примеры интеграции в Visual Studio

Пример: WEB-приложение с мультимодальным ИИ

1. Настройка проекта в Visual Studio:

- Создание решения [ASP.NET](#) Core MVC/Blazor
- Настройка пакетов для работы с ИИ
- Конфигурация аутентификации и авторизации

2. Интеграция различных типов моделей:

- Текстовая аналитика с [ML.NET](#)
- Обработка изображений с ONNX Runtime
- Интеграция с Azure OpenAI через REST API

3. Развертывание и масштабирование:

- Публикация на Azure App Service

- Настройка Azure Functions для бессерверных ИИ-операций
- Мониторинг производительности с Application Insights

Visual Studio предоставляет комплексную среду для создания современных WEB-приложений с гетерогенным ИИ, объединяя мощные инструменты разработки, отладки и развертывания под одной крышей.