



UNIVERSITY
OF TURKU

Laajojen kielimallien käyttö opetuksessa ja ohjelmoinnissa

Juuso Ryttilahti

QR-koodi



UNIVERSITY
OF TURKU





Käsiteltävät aiheet

Yleistä tietoa

Mitä on hallusinaatio

Opettajille

Mahdollisia käyttötapoja

Oppilaille

Miltä tekoäly näyttää oppilaan näkökulmasta?

Huijaamisen helppous



Yleisesti

- Generatiivisen tekoälyn suorituskyky on noussut lyhyessä ajassa huomattavan paljon.
- Tekoäly on tullut jäädäkseen.
- Malleja on useita erilaisia (ChatGPT, Gemini jne.).
- Laajojen kielimallien suurin vahvuus on niiden joustavuus.



Yleisesti #2

- Tekoäly ei osaa oikeasti ajatella.
- Tekoälyn antamien vastauksien laadukkuuteen vaikuttavia seikkoja:
 - Tekstin pituus,
 - esimerkkien lukumäärä, symbolit ja niiden järjestys (jopa $a/b... & i/ii...$)¹,
 - käytetyt sanavalinnat,
 - sattuma (generatiivinen tekoäly on ei-deterministinen)...
- On ehkä helppoa ajatella, että tekoäly on usein kuin automyyjän karikatyyri.

Mitä on hallusinaatio?

- Hallusinaation tarkat määritelmät vaihtelevat. Yleisesti ottaen hallusinaatioilla tarkoitetaan tekoälyn tuottamaa virheellistä tekstiä, joka on usein myös vakuuttavan kuuloista.
- Kaikessa generatiivisen tekoälyn tuottamassa materiaalissa (esim. arviointi) voi olla hallusinaatioita. Ne ovat generatiivisen tekoälyn ominaisuus⁴.

*“Unlike other vendors, however, Lexis+ AI **delivers 100% hallucination-free linked legal citations** connected to source documents, grounding those responses in authoritative resources that can be relied upon with confidence.” (Wellen, 2024a).”*



Opettajan näkökulmasta

- Tekoäly tarjoaa useita mahdollisuuksia ja uhkia.
- Miten integroida tekoäly osana opetusta?
- Entä oppimateriaalien luonti`?

Ota myös huomioon

- Tekijänoikeudet & Tietoturva.



Mahdollisia käyttökohteita

Oppimateriaalien luomisen ja muokkaamisen tukeminen:

- Tehtävien luominen.
- Oppimateriaalin automaattinen kääntäminen ja lokalisointi.
- Rubriikkien luonti.

Opettajaa tukevia toimintoja:

- Tehtävien automaattinen arviointi.
- Tee ohjeistuksesta helppolukuisempi.

Mielenkiintoinen kysymys on, miten tukea oppimista parhaiten?



Tehtävien luominen / Alkuperäinen

The program asks the user for the book's genre (string) and number of pages.

If the book's genre is a **thriller** and it's more than **200 pages** long, a message will be printed saying that you are not in the mood to read it. Otherwise, the book is suitable **for reading**.

Two example executions:

Give a genre: encyclopedia

Give the number of pages: 300

Let's read it

Give a genre: thriller

Give the number of pages: 250

I won't bother



Tehtävien luominen / Variaatio

The program prompts the user to enter the type of music (string) and the duration in minutes.

If the music genre is **rock** and it lasts more than **5 minutes**, a message will be displayed indicating that you're not in the mood to listen to it. Otherwise, the music is suitable **for playing**.

Two sample runs:

Enter the genre: classical

Enter the duration in minutes: 8

I'm not in the mood to listen.

Enter the genre: rock

Enter the duration in minutes: 3

Let's play it.



Rakenteilla oleva työkalu

Generated variations

1/2/3/4/5/6

Modify variation

User:

Write a class 'SingletonDatabase' that ensures only one instance can be created (singleton pattern). Provide a method 'connect(connectionString)' to simulate connecting to a database.

```
class SingletonDatabase {
  // Implementation here
}
```

Copy code

```
class SingletonDatabase {
  static instance;

  constructor() {
    if (SingletonDatabase.instance) {
      return SingletonDatabase.instance;
    }
    SingletonDatabase.instance = this;
    this.connection = null;
  }

  connect(connectionString) {
    if (!this.connection) {
      this.connection = `Connected to ${connectionString}`;
      console.log(this.connection);
    }
  }

  static getInstance() {
    if (!SingletonDatabase.instance) {
      SingletonDatabase.instance = new SingletonDatabase();
    }
    return SingletonDatabase.instance;
  }
}
```

Copy code

User:

Can you make this bit easier, and change the naming of the variables?

Description

Write a class 'SingletonCounter' that ensures only one instance can be created (singleton pattern). Provide a method 'increment()' to increment a counter and 'getValue()' to retrieve the current count.

Skeleton

```
class SingletonCounter {
  // Implementation here
}
```

Copy code

Answer

```
class SingletonCounter {
  static instance;

  constructor() {
    if (SingletonCounter.instance) {
      return SingletonCounter.instance;
    }
    SingletonCounter.instance = this;
    this.count = 0;
  }

  increment() {
    this.count++;
  }

  getValue() {
    return this.count;
  }

  static getInstance() {
    if (!SingletonCounter.instance) {
      SingletonCounter.instance = new SingletonCounter();
    }
    return SingletonCounter.instance;
  }
}
```

Copy code

Description

Write a class 'SingletonConfig' that ensures only one instance can be created (singleton pattern). Provide methods 'setConfig(key, value)' to set a configuration value and 'getConfig(key)' to get a configuration value.

Skeleton

```
class SingletonConfig {
  // Implementation here
}
```

Copy code

Answer

```
class SingletonConfig {
  static instance;

  constructor() {
    if (SingletonConfig.instance) {
      return SingletonConfig.instance;
    }
    SingletonConfig.instance = this;
    this.config = {};
  }

  setConfig(key, value) {
    this.config[key] = value;
  }

  getConfig(key) {
    return this.config[key];
  }

  static getInstance() {
    if (!SingletonConfig.instance) {
      SingletonConfig.instance = new SingletonConfig();
    }
    return SingletonConfig.instance;
  }
}
```

Copy code



Kääntäminen ja lokalisointi

- Tekstin kääntäminen voi olla hankalaa:
 - *"Mieleni tekee jäätelöä"*
 - ChatGPT: *"I feel like having ice cream."*
 - Google translate: *"My mind is making ice cream"*
- Mallit voivat myös lokalisoida materiaalia:
 - TPS -> Arsenal
 - *"Matti Maija Jukka Sanna Anna Arttu Eero Erkki Antero Anne Kari Elina".split(" ");*
 - *-> "John Jane James Jill Ann Arthur Eric Erica Andrew Anne Charles Elizabeth".split(" ");*
- Mallit voivat kääntää hyvin joustavasti eri muodoissa olevaa tekstiä.



Esimerkki

Sen sijaan, että olisimme määritelleet säilöttävän arvon tyyppin, määrittelimme tyyppin generisen tyyppimäärittelyn avulla. Tässä yhteydessä muuttuja TYYPPI siis tarkoittaa, että määrittelemme varsinaisen tyyppin vasta kun luokasta luodaan olio. **Samaa tyyppiä on käytetty myös konstruktorissa ja luokan asetus- ja havainnointimetodeissa. Katsotaan muutama käyttötapaus luokan käytöstä:**

```
public static void main(String[] args) {  
    // kokonaislukutyypin  
    Laatikko<Integer> lukulaatikko = new Laatikko<>(15);  
    System.out.println(lukulaatikko.getArvo());  
    lukulaatikko.setArvo(20 + 30);  
    System.out.println(lukulaatikko.getArvo());  
  
    // merkkijonotyyppinen  
    Laatikko<String> sanalaatikko = new Laatikko<>("Java");  
    System.out.println(sanalaatikko.getArvo());  
    sanalaatikko.setArvo("Moi " + "kaikki" + "!!!");  
    System.out.println(sanalaatikko.getArvo());  
  
    // Opiskelijatyyppinen  
    Opiskelija o = new Opiskelija("Olli", "olli@example.com", 15);  
    Laatikko<Opiskelija> olaatikko = new Laatikko<>(o);  
    System.out.println(olaatikko.getArvo().getNimi());  
}
```

Instead of defining the type of the stored value, we defined the type using a generic type definition. Here, the variable TYPE means that we define the actual type only when an object is created from the class. **The same type has also been used in the constructor and the class's setting and observation methods. Let's look at a few use cases for using the class:**

```
public static void main(String[] args) {  
    // integer type  
    Box<Integer> numberBox = new Box<>(15);  
    System.out.println(numberBox.getValue());  
    numberBox.setValue(20 + 30);  
    System.out.println(numberBox.getValue());  
  
    // string type  
    Box<String> wordBox = new Box<>("Java");  
    System.out.println(wordBox.getValue());  
    wordBox.setValue("Hello " + "everyone" + "!!!");  
    System.out.println(wordBox.getValue());  
  
    // Student type  
    Student s = new Student("Oliver", "oliver@example.com", 15);  
    Box<Student> studentBox = new Box<>(s);  
    System.out.println(studentBox.getValue().getName());  
}
```

Tehtävien arviointi

- Suorituskyky vaihtelee aiheesta, mallista ja kehotteesta riippuen.
- Tarkista aina arvosanaajakauman poikkeama aiempia kurssi-instansseja vasten
 - Jos poikkeaa, niin etsi syy!
- Tarkista myös ns. oudot korrelaatiot (esimerkiksi vastausvaihtoehtojen järjestyksen korrelointi annetun arvosanan kanssa).
- Ihminen validoi aina lopputuloksen!



Ohjeistus / Alkuperäinen

Make this text more understandable

The exercises are a mix of automatically graded ViLLE coding exercises and git repository submissions. Before submitting a repository link, make sure that the repository's visibility is set to public. The repository submissions are reviewed manually by the course personnel, and in those the limit of reviewed submits per exercise is limited to two. That means that after receiving the first score and written feedback you can re-submit the exercise once to improve your answer. Then the second time you receive the points, those will be your final points for the exercise. Naturally, you can request that we take a look at your submission again if you feel that the final score was wrong. However, then we will not take any modifications into account that are made after the second review. On automatically graded exercises, the amount of re-submits is unlimited. Additionally, note that only some of the exercises require concrete implementations, while some are more focused on designing the classes, routines, and specifications for those (don't worry, the specifications are explained in the course).



**UNIVERSITY
OF TURKU**

Ohjeistus / Paranneltu

Here's a clearer version of the text:

The course exercises include both automatically graded ViLLE coding exercises and assignments where you'll submit a link to a Git repository. Before you submit a repository link, make sure the repository is set to public.

The repository submissions are manually reviewed by the course staff, and you can submit each assignment twice for review. After your first submission, you'll receive a score and feedback. You can then improve your work and resubmit it once more. The score you receive after the second submission will be final. If you believe your final score is incorrect, you can ask for another review, but no changes made after the second submission will be considered.

For automatically graded exercises, you can resubmit as many times as you like.

Finally, be aware that some exercises require actual coding implementations, while others focus on designing classes, routines, and specifications. Don't worry—these specifications will be explained during the course.



Oppilaan näkökulmasta

Tekoälyn hyviä puolia:

- Tekoäly voi olla myös väsymätön 24/7 saatavilla oleva tuutori.
- Generatiivinen tekoäly mahdollistaa esimerkiksi projektien laajuuden kasvattamisen.
- Voi mukauttaa selitystä/tehtävää oppilaan taitotason mukaan (adaptiivinen).



Oppilaan näkökulmasta

Tekoälyn huonoja puolia:

- Tekoäly voi antaa liian väärän kuvan omasta osaamisesta.
- Oppilaat voivat myös oppia nojaamaan liikaa tekoälyyn²
- Tasa-arvo: Onko kaikilla yhtäläinen pääsy työkaluihin (mm. kuukausimaksulliset mallit vs. avoimet mallit)?
- Huijaamisen helppous.

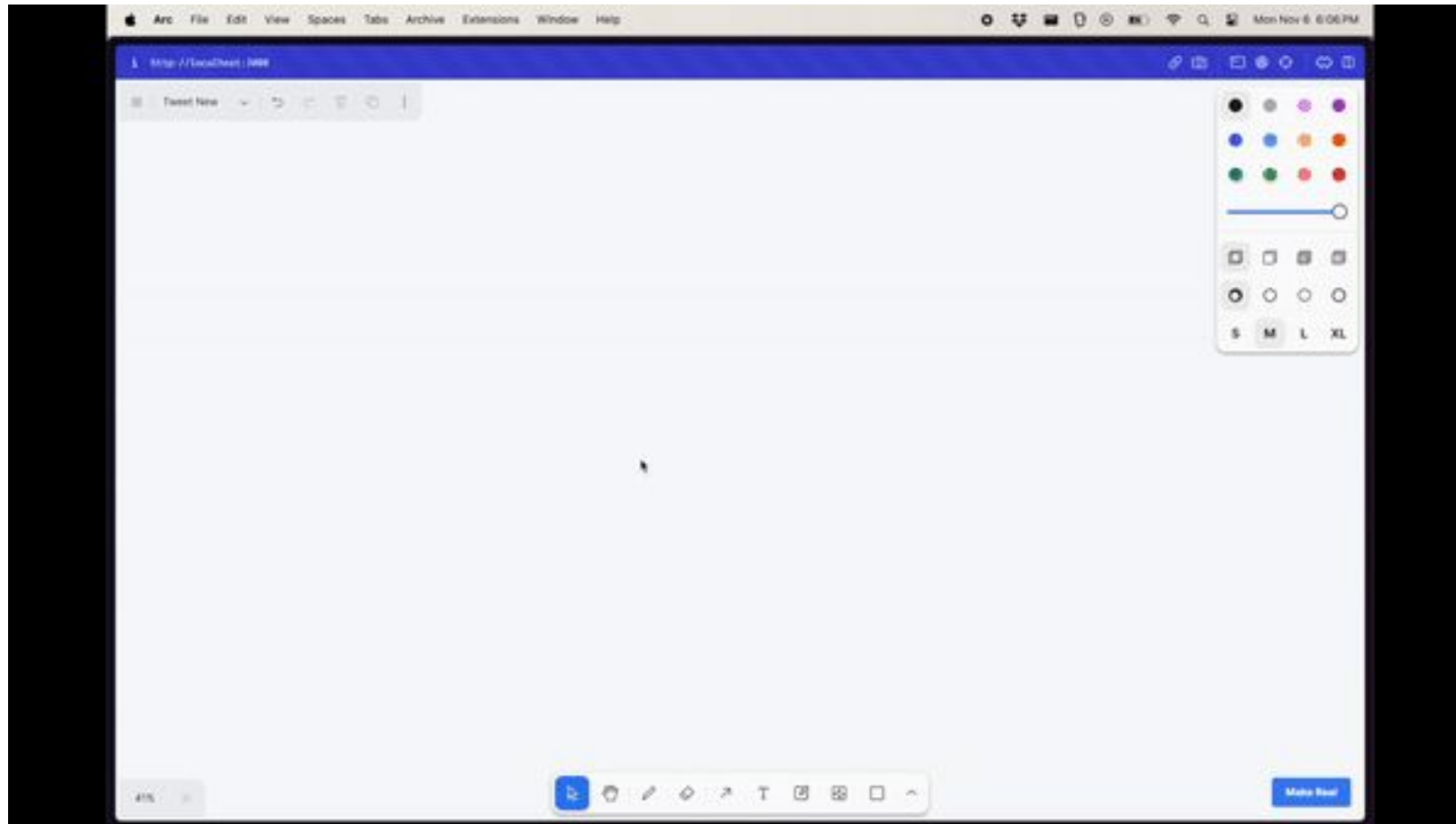


Oppilaan näkökulmasta

- Perusteet on kuitenkin pakko opetella, esim. päässä lasku on hyödyllinen taito edelleen, vaikka kaikilla on nykyään laskin taskussaan...
- Jos oppilaat eivät opi käyttämään tekoälyä, he voivat olla epäedullisessa asemassa myöhemmin työmarkkinoilla.



Nopeasti muuttuva kenttä #1





Nopeasti muuttuva kenttä #2

VISUAL COPILOT

AI-Powered Design to Code

50 - 80% faster Figma to code workflows

Transform designs into production-ready code that aligns with your coding conventions and design system.

AI converts Figma designs to clean code
Use AI to turn structured or unstructured Figma designs into clean, accessible, semantic, and responsive code.
[Learn more >](#)

Chat to iterate the code with AI

Use with or without the Builder Visual CMS

```
Generated Code

import * as React from "react";
import { Hero } from "@/components/Hero";
import { Navigation } from "@/components/Navigation";
import { Button } from "@/components/Button";

function MyComponent() {
  return (
    <Hero
      heading="Portfolio performance tracking made easy"
      supportingText="Reinventing banking for the next-gen finance"
      className="self-stretch"
      navigation={
        <Navigation
          className="self-stretch w-full max-md:max-w-full"
          actions={
            <div className="flex gap-3 self-stretch">
              <Button text="Log in" className="flex-1" primary={false}>
                <Button text="Sign up" className="flex-1" primary={true}>
            </div>
          }
        </Navigation>
      }
    </Hero>
  );
}
```

erience and analyze site
ance with our Cookie



Huijaamisen helppous

- Huijaaminen on helpompaa kuin koskaan
 - Ennen vastaukset kopioitiin kaverilta, nyt voi hyödyntää tekoälyä.
- Tekoälyllä tuotetun tekstin tunnistaminen luotettavasti on luultavasti mahdotonta³.
- Tehtävien muokkaaminen niin, että tekoäly ei osaa vastata kysymykseen on vain väliaikainen ratkaisu.

Kuinka kauan, että selaimiin tulee lisäosa, joka osaa ratkaista tehtävät oppimisalustalla automaattisesti saavutettavuus-kerrosta hyödyntäen (jopa kirjautuneena)?

Visuaalinen tehtävä: tehtävänanto

Oletetaan seuraavanlainen kontekstiton kielioppi

```
< Ohjelma > -> < Lauselistä >
< Lauselistä > -> < Lause >< LauseLista > | < tyhjä >
< Lause > -> < Asetuslause >
< Lause > -> < While-lause >
< Lause > -> < For-lause >
< Lause > -> < Metodikutsu >
< Asetuslause > -> < tunnus >< MathOper >< tunnus >
< While-lause > -> WHILE < Ehto >: < Lauselistä > ENDWHILE
< For-lause > -> FOR (< Asetuslause >; < Ehto >; < Asetuslause >) < Lauselistä > END-
FOR
< Metodikutsu > -> < tunnus > ( < tunnus > )
< Ehto > -> < tunnus >< LooginenOper >< tunnus >
< LooginenOper > -> < | > | <= | >= | == | !=
< MathOper > -> += | -= | *= | /= | =
< tunnus > -> [a-öA-Ö]+ | [0-9]+
```

Minkälainen jäsennyspuu tulee allaolevalle ohjelmalle? Esitä se ASCII puuna

```
sum = 0
FOR (i=0;i<5;i++)
    sum += i
    print(sum)
ENDFOR
```



**UNIVERSITY
OF TURKU**

Visuaalinen tehtävä: GPT-4 vastauksen ASCII-puu

```
<Ohjelma>
|
+- <Lauselist>
  |
  +- <Lause>
    |
    +- <Asetuslause>
      |
      +- <tunnus> "sum"
      +- <MathOper> "="
      +- <tunnus> "0"
    |
    +- <Lauselist>
      |
      +- <Lause>
        |
        +- <For-lause>
          |
          +- <Asetuslause> "i=0"
          +- <Ehto> "i<5"
          +- <Asetuslause> "i++"
          +- <Lauselist>
            |
            +- <Lause>
              |
              +- <Asetuslause>
                |
                +- <tunnus> "sum"
                +- <MathOper> "+="
                +- <tunnus> "i"
            |
            +- <Lauselist>
              |
              +- <Lause>
                |
                +- <Metodikutsu>
                  |
                  +- <tunnus> "print"
                  +- <tunnus> "sum"
          |
          +- <Lauselist> <tyhjä>
```



**UNIVERSITY
OF TURKU**



Yhteenveto

- Mallien suorituskyky on noussut lyhyessä ajassa. Kehitys on edelleen nopeaa.
- Tekoälyllä tuotetun tekstin tunnistaminen luotettavasti on luultavasti mahdotonta.
- Ihmisen kuuluu aina validoida lopputulos.
- Yksinkertaiset jatkokehotteet voivat parantaa lopputulosta merkittävästi.



Viitteet

Yleiset:

- [1] Alzahrani, N., Alyahya, H. A., Alnumay, Y., Alrashed, S., Alsubaie, S., Almushaykeh, Y., ... & Khan, H. (2024). **When benchmarks are targets: Revealing the sensitivity of large language model leaderboards.** arXiv preprint arXiv:2402.01781.
- [2] Bastani, H., Bastani, O., Sungu, A., Ge, H., Kabakçı, O., & Mariman, R. (2024). **Generative ai can harm learning.** Available at SSRN, 4895486.
- [3] Sadasivan, V. S., Kumar, A., Balasubramanian, S., Wang, W., & Feizi, S. (2023). **Can AI-generated text be reliably detected?.** arXiv preprint arXiv:2303.11156.
- [4] Xu, Z., Jain, S., & Kankanhalli, M. (2024). **Hallucination is inevitable: An innate limitation of large language models.** arXiv preprint arXiv:2401.11817.

Tehtävien luominen:

Rytilahti, J., Weerakoon, O., & Kaila, E. (2024). **EXPLORING AI-DRIVEN PROGRAMMING EXERCISE GENERATION.**

Huijaamisen helpous:

Rytilahti, J., & Kaila, E. (2024). **HOW EASY IS IT TO CHEAT?-SOLVING PROGRAMMING EXERCISES AUTOMATICALLY WITH AI.**

Yeadon, W., Inyang, O. O., Mizouri, A., Peach, A., & Testrow, C. P. (2023). **The death of the short-form physics essay in the coming AI revolution.** Physics Education, 58(3), 035027.

Exploratiivinen opas (mm. tehtävien luonti, kääntäminen ja lokalisointi):

Rytilahti, J., Weerakoon, O. 2023. **AI Tools for Study Material Generation** ty.fi/aitools



Diat



UNIVERSITY
OF TURKU

Kiitos!

Ota yhteyttä: jubery@utu.fi



LinkedIn