Tampereen yliopisto

Anniina Honkasaari

SUURET KIELIMALLIT OHJELMISTOKEHITYKSESSÄ

Kandidaatintutkielma
Informaatioteknologian ja viestinnän tiedekunta
Tarkastaja: Tapio Elomaa
Kesäkuu 2024

TIIVISTELMÄ

Anniina Honkasaari: Suuret kielimallit ohjelmistokehityksessä
Kandidaatintutkielma
Tampereen yliopisto
Tietotekniikan tutkinto-ohjelma
Kesäkuu 2024

Ohjelmistokehitys on jatkuvasti kasvava ala, jonka tehokkuutta pyritään lisäämään. Suuret kielimallit eli suurella datamäärällä koulutetut syvät neuroverkkomallit ovat viime vuosina yleistyneet. Niitä voidaan käyttää avuksi ohjelmistokehityksessä, mutta asiaan sisältyy myös riskejä. Tämä työ on kirjallisuuskatsaus aiheeseen. Työssä pyritään vastaamaan kysymykseen "Kuinka suuria kielimalleja on kannattavaa käyttää ohjelmistokehityksessä?".

Suurten kielimallien käyttökohteita ohjelmistokehityksessä ovat muun muassa koodinäytteiden luominen, analyyttiset kysymykset ja reaaliaikainen koodin kommentointi. Nykytekniikalla suuret kielimallit ovat erityisen hyödyllisiä yksinkertaisiin ja rajattuihin ohjelmistokehityksen pyyntöihin.

Suurten kielimallien käytön riskeihin kuuluvat esimerkiksi virheelliset vastaukset, kielimallin koulutusdatan laatu ja tietoturvakysymykset. Jotta ohjelmistokehittäjä voi käyttää turvallisesti ja tehokkaasti suuria kielimalleja, tulee hänen osata tarkastella suurten kielimallien tuottamia vastauksia kriittisesti ja luoda oikeanlaisia käskyjä suurille kielimallille.

Työssä käsitellään myös tutkimusta ChatGPT:n tuottaman ohjelmiston laadusta. Tutkimus on tällä hetkellä niin rajattua, että siitä ei voida vetää suuria johtopäätöksiä. Kaikissa käsitellyissä tutkimuksissa on kuitenkin yhtenevä ajatus siitä, että nykyteknologialla ChatGPT ei osaa tuottaa luotettavasti vastauksia vaikeisiin ohjelmointitehtäviin. Helpommat ohjelmointitehtävät ja niiden laadun vertautuminen ihmisiin tuottaa alustavasti lupaavia tuloksia, mutta tarvitsee jatkotutkimusta.

Työssä selviää, että ohjelmistokehittäjien korvaaminen suurilla kielimalleilla on epätodennäköistä lähitulevaisuudessa. Ohjelmistokehityksen tehostamisen kannalta on kuitenkin järkevää selvittää suurten kielimallien ja ihmisten vahvuudet, ja jakaa ohjelmistokehityksen työtehtäviä näiden mukaan. Nykyisellään suurten kielimallien vahvuudet näyttäisivät olevan yksinkertaiset aikaa vievät tehtävät ja staattinen analyysi, ihmisten vahvuudet taas kontekstiin sitominen, tiedon soveltaminen ja kriittisyys.

Avainsanat: ohjelmistokehitys, suuret kielimallit, riskit, käyttökohteet, ohjelmointi, ChatGPT.

SISÄLLYSLUETTELO

1. JOHDANTO

Tietotekniikka on suuressa osassa jokapäiväistä elämäämme. Sen kehitykseen panostetaan jatkuvasti lisää resursseja: useat maat käyttävät nykyään suuren osan tutkimuksen ja kehityksen investoinneistaan tieto- ja viestintätekniikkaan (Casale et al. 2016). Tietotekniikan osa-alueista tässä työssä keskitytään erityisesti ohjelmistokehitykseen. Ohjelmistokehityksen määritelmä vaihtelee, mutta työssä se määritellään erityisesti ohjelmiston luomisena, eikä esimerkiksi niiden testauksena, ylläpitona tai vaatimusmäärittelynä.

Jatkuvan ohjelmistokehityksen tarpeen myötä on etsitty uusia keinoja tehostaa ohjelmistokehitystä. Samaan aikaan viime vuosina on otettu suuria harppauksia kielen käsittelyssä ja koneoppimisen saralla. Tekoälyä on alettu hyödyntämään erilaisissa tehtävissä ja toimialoissa, niistä yhtenä merkittävänä ohjelmointi. Tekoälyä hyödyntämällä on mahdollista parantaa tuottavuutta, vähentää ihmisvirheitä ja automatisoida tehtäviä. (Coello et al. 2024)

Erityisen suuri harppaus tekoälyn saralla tapahtui, kun OpenAI julkaisi *GPT-mallinsa* (generative pre-trained transformer, uutta tuottava esikoulutettu transformer-malli), joista laajan joukon käytössä on ChatGPT. Se on näyttänyt mahdollisuuksia erityisesti tehtävien automatisointiin, koodin laadun parantamiseen ja ohjelmistokehittäjien neuvomiseen. (Coello et al. 2024) Hun (2023) referoiman UBS:n tutkimuksen mukaan ChatGPT on nopeimmin kasvava kuluttajille tehty sovellus historiassa, saavuttaen sata miljoonaa kuukausittaista käyttäjää vain kaksi kuukautta julkaisun jälkeen.

Kehitys ei kuitenkaan tule ongelmitta. Suuret kielimallit ja niiden käytön leviäminen on tuonut huolia tietoturvauhista tekijänoikeuskysymyksiin. Työssä pyritään vastaamaan kysymykseen “Kuinka suuria kielimalleja on kannattavaa käyttää ohjelmistokehityksessä?”. Työssä käydään läpi suurten kielimallien käytön riskejä ja keinoja riskien minimoimiseen. Lisäksi työssä paneudutaan empiiriseen tutkimukseen ChatGPT:n tuottamasta koodista.

Tutkielmassa selviää, että suuret kielimallit, kuten ChatGPT, eivät vielä toimi itsenäisesti etenkään vaikeiden ohjelmointiongelmien äärellä. Ne voivat kuitenkin tehostaa ohjelmistokehittäjien työtä yksinkertaisissa ja rajatuissa alueissa, joten niiden rooli ohjelmistokehittäjien apuna tullee kasvamaan. Suurten kielimallien käytössä on toisaalta

myös riskejä, kuten virheelliset vastaukset ja tietoturvauhat. Näiden seikkojen vuoksi on todennäköistä, että tulevaisuudessa ohjelmistokehittäjien asiantuntemukseen kuuluu suurten kielimallien käyttäminen turvallisesti ja tehokkaasti. Hyviä taitoja suurten kielimallien hyödyntämiseen ovat muun muassa oikeanlaisten käskyjen muotoileminen ja vastausten kriittinen arviointi.

Tutkielmassa esitellään ensin 3. luvussa mitä suuret kielimallit ovat (3.1), sitten niiden käyttökohteet ohjelmistokehityksessä (3.2), käytön riskit (3.3) sekä riskien minimoiminen (3.4). Tämän jälkeen kappaleessa 4. paneudutaan empiiriseen tutkimukseen ChatGPT:n tuottamasta koodista (4.1), analysoidaan tutkimuksia (4.2) ja kootaan tutkimuksien tuloksista tehtäviä johtopäätöksiä (4.3). Lopuksi 5. kappaleessa keskustellaan suurten kielimallien käytöstä ohjelmistokehityksessä, ja annetaan ajatuksia tulevaisuuteen.

2. TUTKIMUSMENETELMÄ

Tämä työ on kirjallisuuskatsaus suurten kielimallien kannattavaan käyttöön ohjelmistokehityksessä. Tutkielman lähteiden haku on suoritettu helmi-huhtikuussa 2024 käyttäen ensisijaisesti Tampereen yliopiston kirjaston Andor-tietokantaa sekä täydentävästi Google Scholaria. Lähteistä on etsitty tietoa pääasiassa suurten kielimallien käyttämisestä ohjelmistokehityksessä.

Tiedon etsinnässä suurten kielimallien käytöstä ohjelmistokehityksessä käytettiin ohjelmistokehitykseen ja suuriin kielimalleihin lausekkeita. Lauseke, jolla Andorista löydettiin suurin osa lähteistä on “large language models” AND ("software development" OR "coding" OR "software engineering"). Haku rajattiin sisältämään vain vertaisarvioituja julkaisuja, sillä aihe on hyvin ajankohtainen ja siitä liikkuu paljon uutta, huomionhakuista ja vertaisarvioimatonta tietoa. Tällä haulla saatiin 124 tulosta, joiden joukosta etsittiin lähteitä manuaalisesti. Toinen lauseke, jolla löydettiin Andorista monta lähdettä oli empirical AND ChatGPT AND programming. Tämä rajattiin niin ikään vertaisarvioituihin lehtiin.

Google Scholarista etsittiin erityisesti hyvin rajattujen hakujen lähteitä, esimerkiksi hakulausekkeella “does ChatGPT increase programming productivity”. Löydetyt lähteet tarkistettiin tämän jälkeen Andorista, ja sopivat lähteet otettiin mukaan työhön. Joitain lähteitä löydettiin myös muiden lähteiden viittauksista. Yksittäisiä vertaisarvioimattomia lähteitä käytettiin suurten kielimallien nykytilanteen taustoittamiseen.

Työhön valittiin lähteitä, jotka koskettivat suurten kielimallien käyttöä ohjelmistokehityksessä, ChatGPT:n tuottamaa ohjelmistoa tai aiheeseen liittyvää taustatietoa. Lähteiden karsinta ja valinta tehtiin hakutulosten otsikoiden ja tiivistelmien avulla. Lähteitä karsittiin, mikäli ne keskittyivät suurelta osin johonkin työn kannalta epäoleelliseen aiheeseen tai näkökulmaan.

Erityisesti empiirisen tutkimuksen osalta tutkimusten tuloksia tiivistettiin jonkin verran. Niissä pyrittiin kuitenkin säilyttämään olennaiset tulokset sekä mahdollisesti tuloksiin vaikuttavat tekijät, jotta lukija ei saisi harhaanjohtavaa mielikuvaa tutkimusten tuloksista.

3. SUURET KIELIMALLIT OHJELMISTOKEHITYKSESSÄ

Ohjelmistokehitykselle on monta määritelmää, mutta tässä tutkielmassa se määritellään erityisesti ohjelmien tuottamisena. Ohjelmistokehitys on suuri ala, oli kyse sitten uusien ohjelmien luonnista tai vanhojen jatkokehityksestä. Ohjelmistokehittäjät etsivät jatkuvasti keinoja tehostaa ja helpottaa toimintaansa. Yksi viime aikoina pinnalle nousseista keinoista on suuret kielimallit.

Alaluvuissa käsitellään tarkemmin suuria kielimalleja ohjelmistokehityksessä. 3.1 keskittyy suuriin kielimalleihin yleisesti ja toimintaperiaatteisiin niiden takana. 3.2 käsitellään suurten kielimallien käyttökohteita ohjelmistokehityksessä ja ohjelmistokehittäjien tukena. 3.3 käsitellään suurten kielimallien käytön riskejä ohjelmistokehityksessä. 3.4 käydään läpi näiden riskien minimointikeinoja.

3.1 Suuret kielimallit

Suuri kielimalli tarkoittaa syvää neuroverkkomallia, jota on koulutettu suurilla datamäärillä, esimerkiksi kirjoilla, koodilla, artikkeleilla ja internet-sivuilla. Kouluttamisen tavoitteena on, että malli oppisi kohdekielensä taustalla piileviä kaavoja ja suhteita. Näin malli pystyy tuottamaan ihmismäistä ja kieliopillisesti hyvää kieltä, tai syntaksiltaan oikeita koodinäytteitä. (Ozkaya 2023a) Tutkimalla toistuvasti koulutusdatan riippuvuussuhteita saadaan suurten kielimallien tuottama sisältö entistä korkealaatuisemmaksi ja kontekstiin sidonnaisemmaksi. Suurten kielimallien käyttö aloitetaan usein käskyllä, minkä jälkeen iteratiivisesti hiotaan käskyä ja keskustellaan kielimallin kanssa ohjaten sisällöntuottoa. (Coello et al. 2024)

Suuria kielimalleja voidaan käyttää esimerkiksi kääntämiseen, tekstin yhteenvetoon ja kysymyksiin vastaamiseen. Niillä on potentiaalia toimia apuvälineinä monenlaisilla aloilla, jos niiden kouluttamiseen käytetty data on laadukasta. Suurten kielimallien isoin haaste onkin, että vaikka mallin tuottama teksti on usein kieliopillisesti oikein ja voi näyttää vakuuttavalta, se ei aina ole totuudenmukaista. (Ozkaya 2023a)

3.2 Suurten kielimallien käyttökohteita ohjelmistokehityksessä

Suuret kielimallit ovat hyödyllisiä yksinkertaisiin ja rajattuihin ohjelmistokehityksen pyyntöihin. Esimerkiksi koodinpätkien luominen onnistuu suurilta kielimalleilta hyvin, jos niille annetaan selkeät ohjeet ja riittävän rajattu tehtävä. (Ebert & Louridas 2023) Suurten

kielimallien tuottama koodi onkin paljon pinnalla ollut aihe, jota käsitellään tässä työssä tarkemmin luvussa 4.

Ohjelmistokehittäjän apuna suuret kielimallit ovat erityisen hyödyllisiä koodikielen syntaksin muistamiseen tai analyyttisiin kysymyksiin (Perkel 2023). Ohjelmistokehittäjä voi siis vaikkapa keskustella suurten kielimallien kanssa löytääkseen ratkaisuja koodiongelmiin.

Suuria kielimalleja voitaisiin käyttää myös reaaliaikaiseen koodin kommentointiin ohjelmistokehittäjän tukena. Hyvin tehty ja kattavasti koulutettu kielimalli voisi todennäköisesti löytää jo aikaisin sellaisia potentiaalisia virheitä, jotka normaalisti löydettäisiin koodista vasta laajan testauksen aikana (Szabó & Bilicki 2023). Reaaliaikaisen koodipalautteen tuottaminen ohjelmistokehittäjälle on toistaiseksi toiminut huonosti, mutta tämän uskotaan kehittyvän jatkossa. Suuria kielimalleja hyödyntävä koodin kommentointiväline voisi korjata syntaksia ja antaa ehdotuksia koodiin. (Ozkaya 2023a)

Maailmassa on käytössä valtavasti vanhentunutta koodia, jota joudutaan myös ylläpitämään ja jatkokehittämään. Ohjelmistokehittäjille sen päivittäminen on yksin työlästä, mutta suuret kielimallit voisivat auttaa vaihtamalla rajattuja koodinäytteitä toiselle ohjelmointikielelle tai muuten modernisoida vanhaa koodia. (Ozkaya 2023a)

Yksi käyttökohde suurille kielimalleille ohjelmistokehityksessä on koodin dokumentaation luominen. Suuret kielimallit olisivat hyviä tuottamaan esimerkiksi tietyn standardin mukaista dokumentaatiota. (Ozkaya 2023a)

3.3 Riskit suurten kielimallien käytössä ohjelmistokehityksessä

Suurilla kielimalleilla ei usein ole kontekstia tuottamaansa koodiin. Tämä johtaa ongelmiin koodityylien eroavaisuudessa sekä koodin toimivuudessa osana suurempia kokonaisuuksia. Kokonaisuus, jossa on sekä käyttäjän että suuren kielimallin tuottamaa koodia, voi olla virheellinen ja keskenään yhteensopimaton. Silloinkin, kun tällainen kokonaisuus toimii, siitä tulla vaikeasti luettava ja ylläpidettävä. (Coello et al. 2024)

Suuret kielimallit ovat riippuvaisia ihmisen antamasta palautteesta. Tämä voi johtaa puolestaan siihen, että kielimallien kanssa keskusteluun ja niiden opettamiseen menee paljon aikaa ja resursseja. (Coello et al. 2024)

Suuret kielimallit antavat toisistaan vaihtelevia vastauksia. Coello et al. (2024) tutkimuksessa OpenAI:n kielimalli GPT-4 tuotti kaksi erilaista tuotosta vaihtamalla vain pieniä osia annetusta käskystä. Tutkimuksessa molemmat tuotetut sisällöt toimivat halutunlaisesti, mutta näin ei aina tapahdu. Tämän satunnaisuuden vuoksi aina ei tiedetä, onko suurella kielimallilla tuotettu koodi toimivaa vai ei. Ozkaya (2023a) nostaa esiin, että syitä suurten kielimallien tuottamien suositusten takana ei ole mahdollista tutkia. Tämä aiheuttaa sen, että suuret kielimallit saattavat levittävät virheellistä tietoa. Mohammadi et al. (2023) tutkimuksessa viedään tätä ajatusta eteenpäin toteamalla, että tekoäly tekee prosessinhallinnasta vaikeampaa läpinäkymättömyytensä vuoksi.

Koulutusdatalla on valtava vaikutus suurten kielimallien toimintaan. Jos kielimallia koulutetaan esimerkiksi internetissä olevalla koodilla, tästä suuri osa on laadultaan huonoa. Se heijastuu suoraan suurten kielimallien tuottamaan koodiin, joka voi esimerkiksi toimia huonosti isoilla määrillä dataa tai sisältää tietoturvauhkia. (Perkel 2023) Jos taas mallin koulutusmateriaalissa on vinoumia tai virheitä, kielimalli korostaa niitä (Ozkaya 2023a). Koulutusmateriaalin tulee olla laadukasta ja monipuolista. Tämän vuoksi suurten kielimallien koulutusmateriaalin laadun tarkkailu on erittäin tärkeää, mutta käytännössä vaikeaa toteuttaa.

Koulutusdataan liittyvät myös kysymykset koulutusmateriaalin tekijänoikeuksista sekä materiaalin tuottajien yksityisyydestä. Suurten kielimallien kehityksen vuoksi odotetaan tulevan tekniikoita, joilla oman sisällön uudelleenkäyttöä voi hallita, mutta näitä ei vielä ole. (Ozkaya 2023a) Käsillä on kiperiä kysymyksiä esimerkiksi siitä, vahingoittaako avoimen lähdekoodin käyttäminen suurten kielimallien koulutukseen koodin tuottajan tekijänoikeuksia (Wong et al. 2023).

Suurten kielimallien käyttö on myös ympäristöä kuluttavaa. Kielimallien koulutuksessa tarvitaan valtava määrä laskentatehoa, mikä johtaa hiilijalanjäljen kasvuun. Hiilijalanjälkeä voitaisiin rajoittaa paremmilla datankeräys- ja varastointimenetelmillä, mutta sellaisia ei vielä ole pystytty kehittämään. (Ozkaya 2023a) Toisaalta suuri laskentateho tuottaa myös korkeita vaatimuksia tietokoneille, joilla niitä tehdään (Wong et al. 2023).

Tietoturvallisuus on yksi suurten kielimallien käytön riskeistä. Esimerkiksi OpenAI:n ChatGPT ohjeistaa käyttäjiään pidättäytymään jakamasta arkaluonteista tietoa. ChatGPT:lle annetut komennot eivät välttämättä mene suoraan sen käyttöön tai koulutukseen, mutta OpenAI voi nähdä annetut komennot. Ne talletetaan ja niitä saatetaan käyttää tulevaisuudessa GPT-mallien kehittämiseen. Tähän liittyy myös muita riskejä: on mahdollista, että internetissä varastoitavat komennot saataisiin

hakkeroimalla, informaation vuotamisella tai vahingossa jakamisella julkiseen tietoon. (Kshetri 2023) Tutkielman kirjoittamishetkellä OpenAI kertoo sivuillaan (Data Controls FAQ 2024), että se tallentaa kuluttajien datan ja käyttää sitä malliensa kouluttamiseen, jollei tämän sallivaa asetusta laita erikseen pois päältä. Mikäli asetuksen laittaa pois päältä, OpenAI säilyttää dataa 30 päivää, mutta vain valvoakseen keskusteluja väärinkäytön varalta.

Toisenlainen tietoturvariski suurissa kielimalleissa on niiden tuottaman, potentiaalisesti haavoittuvuuksia sisältävän koodin käyttäminen. Suuret kielimallit voivat tuottaa tällaista koodia vahingossa, mutta on myös mahdollista manipuloida kielimalleja tuottamaan kyberhyökkäyksiä altista koodia esimerkiksi syöttämällä sen koulutukseen tahallisesti huonoa dataa. (Wong et al. 2023) Tämä onkin erityisesti tulevaisuudessa todennäköinen suunta kyberturvahyökkäyksien kehittymiselle (Ebert & Louridas 2023).

3.4 Suurten kielimallien käytön riskien minimointi

Suurten kielimallien käytön riskejä voidaan minimoida niiden oikeanlaisilla käyttökohteilla ja -tavoilla. Suurten kielimallien käyttöön vaikuttavat esimerkiksi yleiset säädökset, organisaation ohjeistukset sekä käyttäjän, tässä tapauksessa ohjelmistokehittäjän, tavat. Riskien minimoimiseksi onkin tärkeää, että ohjelmistokehittäjällä on suurten kielimallien käyttämiseen vaadittavia taitoja. Näitä taitoja voi kehittää esimerkiksi koulutuksilla.

Ohjelmistokehittäjän tulee osata tarkastella kriittisesti tekoälyn tuottamia vastauksia. Suurten kielimallien käytössä on keskeistä, että ohjelmistokehittäjä osaa arvioida kielimallin tuottamien vastausten luotettavuutta. (Ozkaya 2023a) Jos ohjelmistokehittäjä ei itse tiedä oikeaa vastausta, hänen tulee tietää vähintään, kuinka varmistaa tekoälyn tuottama vastaus. Tämä on tärkeää paitsi ratkaisujen oikeellisuuden varmistamiseksi, myös kyberturvauhkien estämiseksi. (Ebert & Louridas 2023).

Ohjelmistokehityksessä täytyy huomioida myös kokonaisuus. Tietotekniikan ammattilaisilla tulee olla asiantuntijuutta arvioida tekoälyllä tuotetun koodin sopivuutta kokonaisuuteen. Usein unohtuu, että suurilla kielimalleilla tuotettu koodi vaikuttaa koko koodin rakenteeseen. On tärkeää, että jatkossa jo suunniteltaessa ohjelmaa otetaan huomioon tekoälytyökalujen käyttö ja niiden tuottaman koodin sisällyttäminen. (Ozkaya 2023b)

Toisaalta eräs riskeistä oli myös se, että suurten kielimallien hyödyntäminen vaatii paljon ihmisresursseja. Tätä riskiä voidaan minimoida kehotteiden optimoinnilla (eng. prompt engineering), tekoälylle annettavien käskyjen määrittämisen kehittämisenä. Käskyjen oikeanlainen muotoilu vaikuttaa suuresti siihen, että saadaan käyttäjän toivoma

lopputulos. Oikeanlaisten käskyjen tuottaminen on itsessään haastavaa, koska se vaatii käyttäjältä syvää ymmärrystä käsillä olevasta aiheesta. Käskyn muotoilussa täytyy huomioida tehtävä, johon suuren kielimallin vastausta käytetään, haluttu tuloksen muotoilu, sekä haluttu yksityiskohtien määrä. (Ozkaya 2023b) Ohjelmistokehittäjällä on siis jatkossa syytä olla paitsi ymmärrystä työstettävästä aiheesta, myös taito keskustella suurten kielimallien kanssa päästäkseen haluttuun tulokseen.

Toinen tapa minimoida riskejä on asettaa organisaation sisäisiä ohjeistuksia ja rajoituksia suurten kielimallien käyttöön. Esimerkiksi Gurman (2023) raportoi Bloomberg-lehdessä, että teknologiajätti Samsung kielsi ChatGPT:n käytön saatuaan tietoon, että työntekijä syötti sille luottamuksellista koodia. Useat yritykset ovat tehneet linjauksia tekoälyn käytöstä jo ennakoivasti.

Myös säädökset ovat nousseet julkiseen keskusteluun. Euroopan parlamentissa hyväksyttiin maaliskuussa 2024 maailman ensimmäiset tekoälysäännöt. Osa laista on jo astunut voimaan, ja kaikilta osin lakia aletaan soveltaa 24 kuukauden kuluttua sen voimaantulosta. (Europarlamentti 2024b) Tämä asettaa esimerkiksi ChatGPT:lle avoimuusvaatimuksia sekä vaatimuksia noudattaa EU:n tekijänoikeuslakia (Europarlamentti 2024a).

4. KATSAUS EMPIIRISEEN TUTKIMUKSEEN CHATGPT:N TUOTTAMASTA KOODISTA

ChatGPT on OpenAI:n kehittämä chatbot-kuluttajasovellus. Sillä voi tuottaa monia eri asioita, kuten artikkeleita tai runoutta (Hu 2023). Se pohjautuu GPT-3.5-malliin (Liu et al. 2024). ChatGPT on tämän hetken kehittyneimpiä suuria kielimalleja. Luonnollisesti tällaista sovellusta käytetään myös paljon ohjelmoinnin apuna, ja sen julkaisusta lähtien on tutkittu myös ChatGPT:n mahdollisuuksia tuottaa koodia itsenäisesti. Tutkielmassa keskitytään ChatGPT:hen, sillä siitä löytyi nykyaikaisista suurista kielimalleista eniten keskenään vertailukelpoista, vertaisarvioitua tutkimusta. Tässä kappaleessa käydään läpi neljä empiiristä tutkimusta ChatGPT:n tuottamasta koodista, analysoidaan tutkimuksia, ja käydään läpi tutkimuksista tehtäviä johtopäätöksiä.

4.1 Empiirinen tutkimus ChatGPT:n tuottamasta koodista

Kolmessa käsitellyssä tutkimuksessa käytettiin LeetCode-tehtäviä. Se on suosittu alusta, jota ohjelmistokehittäjät käyttävät valmistautuessaan teknisiin haastatteluihin, ja tutkijat käyttävät tehdessään koodituotantoon liittyviä tutkimuksia (Kuhail et al. 2024). LeetCodea voidaan myös käyttää syötettyjen ratkaisujen arviointiin. Sen arviointi perustuu koodin suorituskykyyn ja muistinkäyttöön. (Nascimento et al. 2023) LeetCoden antamaan arviointiin kuitenkin vaikuttaa myös se, kuinka suuri prosessointikuorma LeetCoden palvelimilla on sillä hetkellä (Nascimento et al. 2023, Kuhail et al. 2023). Yhdessä tutkimuksista käytettiin IEEExtreme programming challenge -haasteita. IEEExtreme on vuosittain järjestettävä kansainvälinen ohjelmointikilpailu, johon osallistuu ohjelmoinnin ammattilaisia ympäri maailman (Koubaa et al. 2023).

Nascimento et al. (2023) tutkimuksessa vertailtiin ohjelmistokehittäjiä ja ChatGPT:tä koodin tuottamisessa, erityisesti suorituskyvyn ja muistitehokkuuden näkökulmasta. Tutkimuksessa verrattiin aloittelevien ohjelmoijien, kokeneiden ohjelmoijien sekä ChatGPT:n tuloksia erilaisten vaikeustasojen LeetCode-tehtävissä. Tutkimuksessa käytettiin 2033 LeetCoden tehtävää, joiden tehtävänannot syötettiin suoraan ChatGPT:lle. Minimoidakseen LeetCoden palvelinten kuormituksen vaikutuksen arviointiin, Nascimento et al. suorittivat samojen vastausten arvioinnit useita kertoja. ChatGPT sai joistain tehtävistä toimivan ratkaisun ensiyrittämällä, toisissa tarvittiin useampia iteraatioita kunnes tällainen löytyi. Tutkimuksessa ei sanota suoraan, kuinka useammat iteraatiot saatiin ChatGPT:ltä.

Nascimento et al. (2023) tutkimuksen tulokset näyttivät, että ChatGPT voi ajoittain tuottaa aloittelevia ohjelmistokehittäjiä parempia tuloksia suorituskyvyn ja muistitehokkuuden suhteen, erityisesti helpoissa ja keskivaikeissa ongelmissa. Toisaalta tutkimuksessa ei löytynyt näyttöä sille, että ChatGPT voisi tuottaa parempaa koodia näiden mittarien suhteen kuin kokeneemmat ohjelmistokehittäjät. Nascimento et al. toteavatkin, että heidän tutkimustuloksensa korostavat tehtävien järkevää jakoa tekoälyn ja ihmisten välillä. Tämä kannustaa yhteistyöhön, jossa ohjelmoijan kokemuksella voidaan hioa tekoälyn tuottamaa sisältöä.

Liu et al. (2024) tutkivat ChatGPT:n tuottamaa koodia erityisesti luotettavuuden ja laadun näkökulmasta. He käyttivät tutkimukseen 2033 LeetCoden tehtävää, joista noin puolet olivat keskivaikeita, noin neljäsosa vaikeita ja toinen neljäsosa helppoja. Tehtävät lajiteltiin myös niiden julkaisuajan perusteella. Liu et al. tekivät ChatGPT:n käyttöön tutkimuksessa käskypohjan, jossa kuvataan ohjelmointitehtävä ja runko halutulle vastaukselle.

Liu et al. (2024) tulosten perusteella ChatGPT saa tuotettua toimivaa ja laadukasta koodia parhaiten helppoihin tehtäviin, sitten keskivaikeisiin ja huonoiten vaikeisiin. Mitä pidempi ChatGPT:n tuottama koodinäyte on, sen todennäköisemmin se sisältää virheitä. Tutkimuksessa selvisi myös, että ChatGPT:n tuottama koodi on usein laadullisesti huonoa. Liu et al. arvioivat tuloksia erilaisilla staattisen analyysin työkaluilla sekä sillä, toimiiko tuotettu koodi, joten kysymykset LeetCoden palvelinten kuormituksesta eivät vaikuttane tulosten paikkansapitävyyteen.

Liu et al. (2024) tutkimuksen mukaan ChatGPT:n tulokset ovat selkeästi heikompia uusissa, tammikuun 2022 jälkeen julkaistuissa tehtävissä. Herää siis kysymys, jota myös Nascimento et al. (2023) tutkimuksessaan pohtivat - osaako ChatGPT ratkaista LeetCode-tehtäviä sen vuoksi, että se osaa soveltaa tietoa, vaiko sen vuoksi, että se on koulutettu niillä tai vastaavilla tehtävillä?

Lisäksi Liu et al. (2024) tutkivat, voiko ChatGPT:n kanssa keskustelemalla parantaa sen tuottamaa koodia. ChatGPT:n kanssa keskustelemalla saatiinkin se tuottamaan uusia, parempia iteraatioita koodista. Liu et al. toteavatkin, että ChatGPT:n käytössä käskyjen optimointi sekä edestakainen keskustelu on tärkeää. Tutkimuksessa selvisi myös, että erityyppisten virheiden korjaamiseen toimivat parhaiten erityyppiset käskyt ChatGPT:lle.

Koubaa et al. (2023) puolestaan tutkivat ChatGPT:n koodituottokykyjä hyödyntäen IEEExtreme programming challenge -haasteita. Haasteita valittiin yhteensä 102 ja ne olivat vaikeustasoltaan vaihtelevia. Tutkimuksessa annettiin ChatGPT:lle käskyjä, jotka sisälsivät myös haasteiden ei-toiminnalliset vaatimukset, kuten muistin käytön ja

suoritusajan. ChatGPT:n tulokset arvioitiin samalla työkalulla ja samojen kriteerien mukaan kuin haasteisiin aiemmin osallistuneiden ihmistiimien tulokset. ChatGPT:n pisteet olivat moninkertaisesti huonommat kuin keskimääräisten ihmisosallistujien. Mitä monimutkaisempia tehtävät olivat, sen suurempi ero oli.

Koubaa et al. (2023) huomasivat myös, että ChatGPT:n ja ihmisten pisteet eivät aina korreloineet. Tämä voi tarkoittaa sitä, että ne tehtävät, jotka ovat helpoimpia ihmisille eivät välttämättä ole helpoimpia ChatGPT:lle ja toisin päin.

Kuhail et al. (2024) puolestaan tutkivat ChatGPT:n tuottamaa koodia 180:lla LeetCoden tehtävällä. He näkivät mahdollisena, että ChatGPT:n ongelmanratkaisukykyihin vaikuttivat aiemmissa tutkimuksissa käytettyjen tehtävien suosio sekä ratkaisujen saatavuus. Tämän vuoksi LeetCodesta valittiin tutkimukseen eri tasoisia, eri tyylisiä ja eri suosiotason (eng. popularity score) tehtäviä. Suosiotaso esittää tiedon siitä, kuinka monta saman aiheen tehtävää on saatavilla LeetCodessa. Kuhail et al. huomioivat myös, että ChatGPT:llä on tietty määrä sanayksiköitä (eng. token), joita se voi prosessoida. Se tarkoittaa käytännössä sitä, että ChatGPT:lle syötetyllä käskyllä on rajattu sisältömäärä. Tutkimuksessa tämä huomioitiin niin, että valikoitiin tehtäviä, joiden ongelmankuvaukset olivat lyhyitä.

Kuhail et al. (2024) tutkimuksessa kopioitiin ja liitettiin LeetCoden kysymys suoraa ChatGPT:lle. Tämä tehtiin yhteensä kolme kertaa per kysymys, kukin kerta eri aikaan ja eri käyttäjältä. Tulokset annettiin LeetCoden arvioitavaksi, ja arvioinnissa huomioitiin vastauksen toimivuus, aikatehokkuus ja muistinkäyttö. Tutkimuksen tulokset näyttivät, että ChatGPT:n tuottamien toimivien vastausten aikatehokkuus ja muistinkäyttö vertautuvat ihmisten ratkaisuihin. Tutkimuksen lopussa tosin todettiin, että koska kyseisiin arvoihin vaikuttavat myös LeetCoden palvelinresurssit, tuloksia pitää käsitellä kriittisesti. Tässäkin tutkimuksessa havaintona oli myös se, että mitä vaikeampi tehtävä on, sen huonommin ChatGPT saa tuotettua kriteerien mukaisesti toimivaa koodia. Tutkimuksessa todettiin myös, että korkeamman suosiotason tehtävillä saatiin suhteessa hieman enemmän toimivia ratkaisuja, mutta tätä ei voida kuitenkaan kutsua merkittäväksi korrelaatioksi.

4.2 Empiirisen tutkimuksen analyysi

Tutkimusten keskinäistä vertailua vaikeuttavat tutkimusten erilaiset tutkimuskysymykset ja lähtöasetelmat. Nascimento et al. (2023) vertaili ohjelmistokehittäjiä ja ChatGPT:tä koodin tuottamisessa suorituskyvyn ja muistitehokkuuden näkökulmasta LeetCode-tehtävissä. Liu et al. (2024) tutkivat ChatGPT:n tuottamaa koodia luotettavuuden ja

laadun näkökulmasta LeetCode-tehtävissä. Kuhail et al. (2024) puolestaan vertasivat ihmisten ja ChatGPT:n tuottamaa koodia toimivuuden, aikatehokkuuden ja muistinkäytön näkökulmasta LeetCode-tehtävissä. Koubaa et al. (2023) tutkivat ChatGPT:n koodintuottokykyjä vaihtelevien vaikeustasojen IEEExtreme-tehtävissä, ja vertasivat niitä ohjelmistokehittäjien tuottamaan koodiin. Yhdessä tutkimuksista (Koubaa et al. 2023) siis oli eri alustalta otettuja tehtäviä kuin muissa. Kolmessa tutkimuksista (Nascimento et al. 2023, Koubaa et al. 2023, Kuhail et al. 2024) ChatGPT:tä verrattiin ihmisiin, yhdessä (Liu et al. 2024) taas ei.

ChatGPT:tä myös käytettiin tutkimuksissa eri tavalla. Nascimento et al. (2023) tutkimuksessa kerrottiin, että joissain tehtävissä ratkaisu toimi ensi yrittämällä, toisista tarvittiin useampia iteraatioita. Sitä, miten eri iteraatiot saatiin, ei kuitenkaan tuoda ilmi tutkimuksessa. Nascimento et al. toisaalta kertovat myös, että he kopioivat ja liittivät LeetCoden tehtävät suoraa ChatGPT:lle. Niin ikään Kuhail et al. (2024) tutkimuksessa kopioitiin ja liitettiin LeetCoden kysymykset suoraan ChatGPT:lle, mutta tämä tehtiin kolme kertaa per kysymys, eri aikoihin ja eri käyttäjiltä. Liu et al. (2024) puolestaan pyysivät ChatGPT:ltä vastauksia pohjalla, jossa kuvattiin tehtävä ja sen vaatimukset. He tutkivat myös sitä, voiko ChatGPT:n kanssa keskustelemalla luoda parempia iteraatioita koodista. Koubaa et al. (2023) tutkimuksessa ChatGPT:lle taas annettiin käskyjä, jotka sisälsivät myös tiedot haasteisten ei-toiminnallisista vaatimuksista.

Nascimento et al. (2023) tutkimuksessa ChatGPT:tä vertailtiin ihmisiin. Tutkimukseen valitut ihmisratkaisut olivat LeetCodeen syötettyjä, ja he huomioivatkin, ettei voi olla täyttä varmuutta siitä, että näistä kaikki olisivat varmasti ihmisten ratkaisuja. On mahdollista, että osa alustaan syötetyistä ratkaisuista on vähintään tekoälyn avustuksella tuotettuja.

ChatGPT:n kanssa tehtävää tutkimusta vaikeuttavat myös satunnaisuuselementit - koska prosessi on läpinäkymätön ja siihen vaikuttavat neuroverkot, on epätodennäköistä saada tismalleen samoja tuloksia kerrasta toiseen. Kuhail et al. (2024) käyttämät usean toistokerran tutkimusmenetelmät voisivat sopia tulevissa tutkimuksissa tämän ongelman mitätöimiseen. Myös Liu et al. (2024) huomioivat tämän heikkouden, ja käyttivät suurta määrää tehtäviä arviointiin. Jokaisen arviointiin käytettiin ensimmäistä ChatGPT:n tuottamaa vastausta. He myös asettivat ChatGPT:n *lämpötilaparametrin* nollaan, joka heidän mukaansa saa ChatGPT:n tuottamaan samanlaisia vastauksia samoille käskyille. Yksinkertaistettuna lämpötilaa voisi kutsua mallin luovuudeksi. Lämpötilaparametrin käyttöä voisi tutkia jatkossa enemmän, jotta selviäisi, kuinka paljon sillä saataisiin rajoitettua vastausten vaihtelevuutta.

Kuhail et al. (2024) tutkimuksessa huomioitiin ChatGPT:n sanayksikköraja, joka saattaa vaikuttaa myös muiden tutkimusten luotettavuuteen. Sanayksikköraja tarkoittaa sitä, että ChatGPT:lle syötetyt ongelmankuvaukset eivät voi sisältää tiettyä määrää enempää informaatiota. Toinen ongelma, jota täytyy ottaa huomioon tämänkaltaisten tutkimusten luotettavuudessa, nousi esiin sekä Liu et al. (2024) että Nascimento et al. (2023) tutkimuksissa: ratkaisujen helppo saatavuus. Onko ChatGPT:tä voitu opettaa samankaltaisilla tai jopa samoilla tehtävillä, kuin tutkimuksissa käytetyt tehtävät?

Ihmisiin verratessa nousee ongelmia näkökulmien kanssa. Nascimento et al. (2023) kirjoittivat, että jos he olisivat niputtaneet ohjelmoijat yhteen eivätkä erotelleet heitä aloitteleviin ja kokeneisiin, voitaisiin tutkimuksessa tehdä johtopäätös siitä, että ChatGPT pärjää paremmin kuin ihmiskoodaajat. Täysin luotettavia ja objektiivisia määritelmiä ohjelmoijien taitoluokituksille on vaikea löytää - Nascimento et al. luokittelivat koodaajat taitotasoihin neljän tehtävän perusteella. Tämä herättää kysymyksen ihmisiin vertaamisen mielekkyydestä - toisaalta tarpeeksi laajalla näkökulmalla voidaan saada perspektiiviä suurten kielimallien tuottaman koodin tasoon, toisaalta ei ole välttämättä mielekästä koittaa löytää ChatGPT:lle tarkkaa sijoitusta suhteessa eritasoisiin ihmiskoodaajiin, sillä tasojen tarkka määrittely on vaikeaa.

4.3 Johtopäätökset empiirisestä tutkimuksesta

Kaikki tutkimukset tukivat ajatusta siitä, että mitä vaikeampi kooditehtävä, sitä huonompia ChatGPT:n tuottamat tulokset. Muutoin tutkimusten tulokset vaihtelivat hieman, mutta niin toisaalta myös niiden tarkoitukset ja keinot.

Kahdessa tutkimuksista, Nascimento et al. (2023) sekä Kuhail et al. (2024), saatiin tuloksia siitä, että ChatGPT saa tuotettua parempia tuloksia kuin vähintään aloitteleva ohjelmistokehittäjä. Nascimento et al. tarkasteli asiaa suorituskyvyn ja muistitehokkuuden näkökulmasta, Kuhail et al. toimivuuden, aikatehokkuuden ja muistinkäytön näkökulmasta. Kuhail et al. huomio siitä, että tuloksiin voi vaikuttaa LeetCoden palvelinresurssit tekevät kuitenkin tästä johtopäätöksestä nykyisellään huteran, ja se vaatii lisää tutkimusta.

Nascimento et al. (2023) tutkimuksessa todetaan, ettei löytynyt todisteita ChatGPT:n tuottaman koodin paremmuudesta suhteessa kokeneisiin ohjelmistokehittäjiin. Koubaa et al. (2023) tuloksista puolestaan nähdään, että ChatGPT:n tuottama koodi oli ainakin IEEExtreme-arviointikriteereissä moninkertaisesti ihmisten tuottamaa koodia huonompaa. Liu et al. (2024) puolestaan toteaa, että ChatGPT:n tuottamassa koodissa

on usein laadullisia virheitä. Tästä voidaan päätellä, että ChatGPT ei pysty nykyteknologialla korvaamaan ohjelmistokehittäjiä, tai ainakaan kokeneempia sellaisia.

Toisaalta Koubaa et al. (2023) huomasi, että ChatGPT:n ja ihmisten pisteet eivät aina korreloineet keskenään. Liu et al. (2024) tutkimuksessa taas iteroitiin koodia ChatGPT:n kanssa keskustelemalla, ja saatiin tuotettua parempia versioita koodista. Nascimento et al. (2023) puolestaan totesivat, että heidän tutkimustuloksensa korostavat tehtävien järkevää jakoa tekoälyn ja ihmisen välillä. Nämä ajatukset tukevat toisiaan: sekä suurille kielimalleille että ihmiselle on oma roolinsa. ChatGPT:llä ja ohjelmoijilla on siis omat vahvuutensa, mutta niiden tarkka määrittäminen vaatii syvempää tutkimusta aiheeseen.

Liu et al. (2024) tutkimuksessa selvisi myös, että erityyppiset käskyt toimivat parhaiten, jotta saadaan ChatGPT korjaamaan erityyppisiä virheitä sen tuottamassa koodissa. Tämä ajatus tukee kehotteiden optimoinnin tärkeyttä: käskyjen muotoilu on kriittinen osa suurten kielimallien käyttöä. Tämä huomioitiin myös muissa tutkimuksissa (Nascimento et al. 2023, Koubaa et al. 2023), joissa todettiin, että muotoilu voi suurestikin vaikuttaa ChatGPT:n tuloksiin.

5. KESKUSTELU

Suurten kielimallien käyttökohteissa ohjelmistokehityksessä korostuvat yksinkertaisuus, suoraviivaiset kysymykset ja vähäinen kielimallin suorittama tiedon soveltaminen. Työssä käytetystä kirjallisuudesta tulee vahvasti ilmi suurten kielimallien heikkous kontekstiin sitomisessa ja isojen kokonaisuuksien hahmottamisessa. Tutkimusten perusteella jopa yksi edistyneimmistä suurista kielimalleista, ChatGPT, pärjää huonosti vaikeampien tehtävänantojen kanssa. On siis reilua sanoa, että suuret kielimallit eivät ainakaan lähitulevaisuudessa tule korvaamaan ohjelmistokehittäjiä. Toisaalta lähteistä Ebert ja Louridas (2023) esittävät erilaisen näkökulman todeten, että suurten kielimallien vuoksi on vaikeaa kuvitella tulevaisuutta, jossa ohjelmistokehittäjät ovat yhtä korkeasti palkattuja kuin nykyään.

Myöskään suurten kielimallien tuottamien suurten ohjelmistojen tulevaisuus ei näytä kaikkien lähteiden mukaan toiveikkaalta. Pengin (2023) mukaan tuottaaksemme nykytekniikalla suuria ohjelmia alusta loppuun suurilla kielimalleilla ensimmäinen rajoittava tekijä on se, että suuri kielimalli tarvitsee ohjelmistokehittäjältä useita selkeitä askeleita saavuttaakseen lopputuloksen. Tämä tarkoittaa sitä, että ohjelmistokehittäjän tulisi ymmärtää täysin suurilta kielimalleilta pyydettävän ohjelman rakenne ja tarkka toteutusprosessi, mikä ei ole realistista.

Onkin tämän pohjalta todennäköistä, että suuret kielimallit tulevat kasvattamaan rooliaan nimenomaan ohjelmistokehittäjien apuna. Jo nyt ohjelmistokehittäjät voivat hyödyntää suuria kielimalleja koodisyntaksiin liittyvissä asioissa ja ohjelmistokehityksen ongelmanratkaisussa. Myös suurten kielimallien tuottamia apuohjelmia kehitettäneen jatkossa.

Kappaleessa 3.4 käytiin läpi suurten kielimallien käytön tuottamien riskien minimointia. Keinoissa korostui erityisesti ohjelmistokehittäjän rooli suurten kielimallien käyttäjänä. Ohjelmistokehittäjän rooli tuleekin jatkossa muuttumaan suurten kielimallien hyödyntämisen myötä (Mohammadi et al. 2023). Ohjelmistokehittäjän tulee jatkossa kyetä korostamaan ihmisen vahvuuksia verrattuna suuriin kielimalleihin: kokonaisuuden hahmottaminen ja kriittinen ajattelu. Esimerkiksi kehotteiden optimointi tulee olemaan ohjelmistokehittäjälle jatkossa tärkeä taito (Mohammadi et al. 2023).

Ohjelmistokehityksen eettistä puolta on tärkeää jatkossa korostaa. Tulevaisuus riippuu siitä, voidaanko ohjelmistotuotanto tehdä mahdollisimman korkealla eettisellä tasolla ja

vastuullisilla käytännöillä. (Ozkaya 2023a) Nähtäväksi jää, pysyykö suurten kielimallien käytön eettinen keskustelu nopean kehityksen mukana.

Suurten kielimallien saama huomio on kasvanut rajusti viime vuosina. Erityisesti uusia teknologioita ei olla ehditty tieteellisesti tutkia laajasti. Vielä vähemmän tutkimusta aiheesta on ehtinyt kulkea vertaisarviointiprosessien läpi, rajoittaen entisestään aiheesta löytyvää luotettavaa tietoa. Tulevaisuudessa suurista kielimalleista saataneen laajempaa kuvaa, jonka pohjalta voidaan nähdä tarkemmin suurten kielimallien roolia jatkossa.

Tutkielmassa käytetty kirjallisuus on vertaisarvioitua, mutta valtaosa siitä on julkaistu vuosina 2023–2024. Koska tutkielma on tehty keväällä 2024, on kirjallisuuden laadussa riskejä. Vaikka tutkimus on vertaisarvioitu, voidaan silti julkaisun jälkeen kritisoida sitä laajasti tai todistaa siinä esitettyjä väitteitä vääräksi. Koska tutkielmassa käytetty kirjallisuus on uutta, tällaisten kritiikkien esiin tuloon ei ole ollut juurikaan aikaa.

Tämän tutkielman näkökulma on rajattu lähinnä ohjelmiston tekoon. Tämän reuna-alueilla on paljon lisää huomioon otettavaa tietoa, kuten ohjelmistojen testaus ja suunnittelu. Entistä täydemmän kuvan ohjelmiston tekoon liittyvistä seikoista saisi laajentaen aihetta koskettamaan myös muita ohjelmistoihin liittyviä asioita.

Tutkielmassa on keskitytty myös suurista kielimalleista erityisesti ChatGPT:hen. Muita samankaltaisia työkaluja on olemassa, esimerkiksi koodauksen aputyökalu GitHub Copilot. Vastaavat työkalut ovat niin ikään uusia, ja vähemmän suosittuja kuin valtavan suosion ChatGPT, joten niistä ei ole vielä laajasti vertaisarvioitua tutkimusta. Jatkossa eri suuriin kielimalleihin pohjautuvien työkalujen vastauksia voisi verrata keskenään, ja ylipäätään tutkia tarkemmin myös muiden työkalujen laatua ja potentiaalia.

6. YHTEENVETO

Ohjelmistoa tuotetaan jatkuvasti valtavia määriä, ja tämän vuoksi kaikki mahdollisesti tuotannon tehostukseen käytettävä herättää paljon mielenkiintoa. Suuria kielimalleja voidaan tällä hetkellä käyttää ohjelmistokehittäjän työtä helpottavina apukeinoina.

Suurten kielimallien vahvuus näyttääkin olevan toimia ihmisten kanssa yhteistyössä. Niiden hyödyntäminen keskittyykin erityisesti esimerkiksi pienten koodinpätkien luontiin ja analyyttisiin kysymyksiin vastaamiseen.

Suuret kielimallit, kuten ChatGPT, voivat tuottaa toimivaa koodia yksinkertaisilla tehtävänannoilla. Tarvitaan kuitenkin lisää tutkimusta aiheesta, jotta niiden tuottaman ohjelmiston tasosta voidaan päästä lopulliseen selvyyteen. Vaikeita ohjelmointitehtäviä tämänhetkinen suurten kielimallien teknologia ei pysty ratkaisemaan luotettavalla tasolla.

Suurten kielimallien käytössä on myös virheellisiin vastauksiin liittyviä riskejä. Jotta nykyaikainen ohjelmistokehittäjä voi turvallisesti käyttää suuria kielimalleja ohjelmoinnin apuna, on hänen oltava kriittinen sen tuottamia vastauksia kohtaan. Toisaalta myös käskyjen suunnittelu vaikuttaa vastausten laatuun paljon, ja kehotteiden optimointi tullee nousemaan pinnalle ohjelmistokehittäjien työnkuvissa.

LÄHTEET

Casale, G., Chesta, C., Deussen, P., Di Nitto, E., Gouvas, P., Koussouris, S., Stankovski, V., Symeonidis, A., Vlassiou, V., Zafeiropoulos, A., & Zhao, Z. (2016). Current and Future Challenges of Software Engineering for Services and Applications. Procedia Computer Science, Vol.97, pp. 34–42. https://doi.org/10.1016/j.procs.2016.08.278

Coello, C. E. A., Alimam, M. N., & Kouatly, R. (2024). Effectiveness of ChatGPT in Coding: A Comparative Analysis of Popular Large Language Models. Digital, Vol.4(1), pp. 114–125. https://doi.org/10.3390/digital4010005

Ebert, C. & Louridas, P. (2023). Generative AI for Software Practitioners. IEEE Software, Vol.40(4), pp. 30–38. https://doi.org/10.1109/MS.2023.3265877

Europarlamentti (2024a). EU:n tekoälysäädös on ensimmäinen laatuaan. Euroopan parlamentti. Päivitetty 13.3.2024. Saatavissa (viitattu 29.4.2024): https://www.europarl.europa.eu/topics/fi/article/20230601STO93804/eu-n-tekoalysaados-on-ensimmainen-laatuaan

Europarlamentti (2024b). Parlamentti hyväksyi maailman ensimmäiset tekoälysäännöt. Euroopan parlamentti. Päivitetty 13.3.2024. Saatavissa (viitattu 29.4.2024): https://www.europarl.europa.eu/news/fi/press-room/20240308IPR19015/parlamentti-hyvaksyi-maailman-ensimmaiset-tekoalysaannot

Gurman, M. (2023). Samsung Bans Staff's AI Use After Spotting ChatGPT Data Leak. Bloomberg. Päivitetty 2.5.2023. Saatavissa (viitattu 24.4.2023): https://www.bloomberg.com/news/articles/2023-05-02/samsung-bans-chatgpt-and-other-generative-ai-use-by-staff-after-leak?embedded-checkout=true

Hu, K. (2023). ChatGPT sets record for fastest-growing user base - analyst note. Reuters. Päivitetty 2023. Saatavissa (viitattu 2.4.2023): https://www.reuters.com/technology/chatgpt-sets-record-fastest-growing-user-base-analyst-note-2023-02-01

Koubaa, A., Qureshi, B., Ammar, A., Khan, Z., Boulila, W., & Ghouti, L. (2023). Humans are still better than ChatGPT: Case of the IEEEXtreme competition. Heliyon, Vol.9(11), e21624. https://doi.org/10.1016/j.heliyon.2023.e21624

Kshetri, N. (2023). Cybercrime and Privacy Threats of Large Language Models. IT Professional, Vol.25(3), pp. 9–13. https://doi.org/10.1109/MITP.2023.3275489

Kuhail, M. A., Mathew, S. S., Khalil, A., Berengueres, J., & Shah, S. J. H. (2024). “Will I Be Replaced?” Assessing ChatGPT’s Effect on Software Development and Programmer Perceptions of AI Tools. Science of Computer Programming, Vol.235, 103111. https://doi.org/10.1016/j.scico.2024.103111

Liu, Y., Le-Cong, T., Widyasari, R., Tantithamthavorn, C., Li, L., Le, Xuan-Bach D., & Lo, D. (2024). Refining ChatGPT-Generated Code: Characterizing and Mitigating Code Quality Issues. ACM Transactions on Software Engineering and Methodology. https://doi.org/10.1145/3643674

Mohammadi, H., Ghardallou, W., Brick, E., & Mili, A. (2023). On the persistent rumors of the programmer's imminent demise. Software and Systems Modeling, Vol.22, pp. 1969–1976. https://doi.org/10.1007/s10270-023-01136-y

Nascimento N., Alencar P., & Cowan D. (2023). Artificial Intelligence vs. Software Engineers: An Empirical Study on Performance and Efficiency using ChatGPT. Proceedings of Conference of the Center for Advanced Studies on Collaborative Research. https://dl.acm.org/doi/10.5555/3615924.3615927

Data Controls FAQ, OpenAI. Päivitetty 4/2024. Saatavissa (viitattu 24.4.2024): https://help.openai.com/en/articles/7730893-data-controls-faq

Ozkaya, I. (2023a) Application of Large Language Models to Software Engineering Tasks: Opportunities, Risks, and Implications. IEEE software, Vol.40(3), pp. 4–8. https://doi.org/10.1109/MS.2023.3248401

Ozkaya, I. (2023b). Can Architecture Knowledge Guide Software Development With Generative AI? IEEE Software, Vol.40(5), pp. 4–8. https://doi.org/10.1109/MS.2023.3306641

Peng, X. (2023). Software development in the age of intelligence: embracing large language models with the right approach. Frontiers of Information Technology & Electronic Engineering, Vol.24, pp. 1513–1519. https://doi.org/10.1631/FITEE.2300537

Perkel, J. M. (2023). Six tips for better coding with ChatGPT. Nature, Vol.618, pp. 422–423. https://doi.org/10.1038/d41586-023-01833-0

Szabó, Z., & Bilicki, V. (2023). A New Approach to Web Application Security: Utilizing GPT Language Models for Source Code Inspection. Future Internet, Vol.15(10), pp. 326. https://doi.org/10.3390/fi15100326

Wong, M. F., Guo, S., Hang, C. N., Ho, S. W., & Tan, C. W. (2023). Natural Language Generation and Understanding of Big Code for AI-Assisted Programming: A Review. Entropy, Vol.25(6), pp. 888. https://doi.org/10.3390/e25060888