



(12) **Veröffentlichung**

der internationalen Anmeldung mit der  
(87) Veröffentlichungs-Nr.: **WO 2019/002996**  
in der deutschen Übersetzung (Art. III § 8 Abs. 2  
IntPatÜG)  
(21) Deutsches Aktenzeichen: **11 2018 001 952.7**  
(86) PCT-Aktenzeichen: **PCT/IB2018/054034**  
(86) PCT-Anmeldetag: **06.06.2018**  
(87) PCT-Veröffentlichungstag: **03.01.2019**  
(43) Veröffentlichungstag der PCT Anmeldung  
in deutscher Übersetzung: **20.02.2020**

(51) Int Cl.: **G06F 3/048 (2013.01)**

(30) Unionspriorität:  
**15/634,451**      **27.06.2017**      **US**  
  
(71) Anmelder:  
**International Business Machines Corporation,**  
**Armonk, N.Y., US**  
  
(74) Vertreter:  
**Richardt Patentanwälte PartG mbB, 65185**  
**Wiesbaden, DE**

(72) Erfinder:  
**Watson, Patrick, Yorktown Heights, NY, US; Ahn,**  
**Jae-Wook, Yorktown Heights, NY, US; Chang,**  
**Maria, Yorktown Heights, NY, US; Sundararajan,**  
**Sharad, Chandra, Yorktown Heights, NY, US**

Prüfungsantrag gemäß § 44 PatG ist gestellt.

**Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen.**

(54) Bezeichnung: **VERBESSERTES VISUELLES DIALOGSYSTEM FÜR INTELLIGENTE TUTOREN**

(57) Zusammenfassung: Systeme, Verfahren und Computerprogrammprodukte zum Ausführen einer Operation, aufweisend ein Empfangen einer Texteingabe über eine Chat-Schnittstelle einer Tutor-Anwendung, ein Identifizieren eines Konzepts in der Texteingabe durch mindestens einen Klassifizierer, der auf die Texteingabe angewendet wird, ein Zuordnen des Konzepts in der Texteingabe auf mindestens eine visuelle Aktion oder ein erstes visuelles Objekt, ein Generieren, auf Grundlage eines ersten Maschinlern- (ML) Modells, einer ersten Programmcodeanweisung, die mindestens der einen visuellen Aktion oder dem ersten visuellen Objekt entspricht, und ein Ausführen der ersten Programmcodeanweisung zum Modifizieren einer Visualisierungsschnittstelle der Tutor-Anwendung auf Grundlage der Texteingabe, die über die Chat-Schnittstelle empfangen wurde.

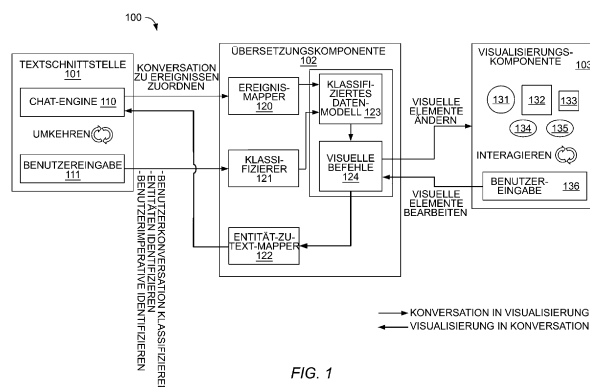


FIG. 1

**Beschreibung****HINTERGRUND**

**[0001]** Die vorliegende Erfindung betrifft Computer-Software und insbesondere Computer-Software, die eine verbesserte visuelle Dialogschnittstelle für intelligente Tutoren bereitstellt.

**[0002]** Intelligente Tutoriensysteme stellen für Benutzer Lernerfahrungen bereit, ohne ein Eingreifen durch einen menschlichen Lehrer zu erfordern. Einige intelligente Tutoren stellen eine Datenübertragungsschnittstelle auf Textgrundlage bereit, die es Benutzern ermöglicht, mit dem intelligenten Tutor per Text zu chatten (und umgekehrt). Andere Komponenten (wie beispielsweise grafische Lernerflächen) des intelligenten Tutor-Systems sind jedoch von den Benutzer-Datenübertragungen auf Textgrundlage isoliert. Desgleichen ist die Datenübertragungsschnittstelle auf Textgrundlage von der Interaktion isoliert, wenn die Benutzer mit anderen Komponenten des Systems interagieren.

**KURZDARSTELLUNG DER ERFINDUNG**

**[0003]** In einer Ausführungsform der Erfindung weist ein Verfahren ein Empfangen einer Texteingabe über eine Chat-Schnittstelle einer Tutor-Anwendung auf, ein Identifizieren eines Konzepts in der Texteingabe durch mindestens einen Klassifizierer, der auf die Texteingabe angewendet wird, ein Zuordnen des Konzepts in der Texteingabe zu mindestens einer visuellen Aktion oder einem ersten visuellen Objekt, ein Generieren, auf Grundlage eines ersten Maschinenlern- (ML) Modells, einer ersten Programmcodeanweisung, die mindestens der einen visuellen Aktion oder dem ersten visuellen Objekt entspricht, und ein Ausführen der ersten Programmcodeanweisung zum Modifizieren einer Visualisierungsschnittstelle der Tutor-Anwendung auf Grundlage der Texteingabe, die über die Chat-Schnittstelle empfangen wurde.

**[0004]** In einer weiteren Ausführungsform der Erfindung weist ein System einen Prozessor und einen Arbeitsspeicher auf, der Anweisungen speichert, die, wenn sie durch den Prozessor ausgeführt werden, eine Operation ausführen, die ein Empfangen einer Texteingabe über eine Chat-Schnittstelle einer Tutor-Anwendung aufweist, ein Identifizieren eines Konzepts in der Texteingabe durch mindestens einen Klassifizierer, der auf die Texteingabe angewendet wird, ein Zuordnen des Konzepts in der Texteingabe zu mindestens einer visuellen Aktion oder einem ersten visuellen Objekt, ein Generieren, auf Grundlage eines ersten Maschinenlern-(ML) Modells, einer ersten Programmcodeanweisung, die mindestens der einen visuellen Aktion oder dem ersten visuellen Objekt entspricht, und ein Ausführen der

ersten Programmcodeanweisung zum Modifizieren einer Visualisierungsschnittstelle der Tutor-Anwendung auf Grundlage der Texteingabe, die über die Chat-Schnittstelle empfangen wurde.

**[0005]** In einer weiteren Ausführungsform der Erfindung ist in einem durch einen Computer lesbaren Speichermedium ein durch einen Computer lesbarer Programmcode verkörpert, wobei der durch einen Computer lesbare Programmcode durch einen Prozessor ausführbar ist, um eine Operation auszuführen, die ein Empfangen einer Texteingabe über eine Chat-Schnittstelle einer Tutor-Anwendung aufweist, ein Identifizieren eines Konzepts in der Texteingabe durch mindestens einen Klassifizierer, der auf die Texteingabe angewendet wird, ein Zuordnen des Konzepts in der Texteingabe zu mindestens einer visuellen Aktion oder einem ersten visuellen Objekt, ein Generieren, auf Grundlage eines ersten Maschinenlern- (ML) Modells, einer ersten Programmcodeanweisung, die mindestens der einen visuellen Aktion oder dem ersten visuellen Objekt entspricht, und ein Ausführen der ersten Programmcodeanweisung zum Modifizieren einer Visualisierungsschnittstelle der Tutor-Anwendung auf Grundlage der Texteingabe, die über die Chat-Schnittstelle empfangen wurde.

**Figurenliste**

**Fig. 1** veranschaulicht beispielhafte Komponenten eines intelligenten Tutor-Systems, das eine verbesserte visuelle Dialogschnittstelle gemäß einer Ausführungsform der Erfindung bereitstellt.

**Fig. 2A bis Fig. 2D** stellen Beispiele für verbesserte visuelle Dialogschnittstellen für intelligente Tutoren gemäß verschiedenen Ausführungsformen der Erfindung dar.

**Fig. 3** ist ein Ablaufplan, der ein beispielhaftes Verfahren zum Bereitstellen einer verbesserten visuellen Dialogschnittstelle für intelligente Tutoren gemäß einer Ausführungsform der Erfindung veranschaulicht.

**Fig. 4** ist ein Ablaufplan, der ein beispielhaftes Verfahren zum Schulen von Klassifizierern gemäß einer Ausführungsform der Erfindung veranschaulicht.

**Fig. 5** ist ein Ablaufplan, der gemäß einer Ausführungsform der Erfindung ein beispielhaftes Verfahren zum Übersetzen einer Texteingabe zu einem Generieren einer Anweisung zum Modifizieren einer Visualisierung veranschaulicht.

**Fig. 6** ist ein Ablaufplan, der gemäß einer Ausführungsform der Erfindung ein beispielhaftes Verfahren zum Übersetzen einer Benutzereingabe veranschaulicht, die an einer Visualisie-

rungsschnittstelle empfangen wird, um eine Anweisung in natürlicher Sprache zu erstellen.

**Fig. 7** ist ein Blockschaubild, das ein beispielhaftes System veranschaulicht, das eine verbesserte visuelle Dialogschnittstelle für intelligente Tutoren gemäß einer Ausführungsform der Erfindung bereitstellt.

#### AUSFÜHRLICHE BESCHREIBUNG

**[0006]** Hierin offenbarte Ausführungsformen der Erfindung stellen ein verbessertes intelligentes Tutor-System bereit, das eine Datenübertragung zwischen Benutzern und verschiedenen Komponenten des intelligenten Tutors verbessert. Wenn Benutzer zum Beispiel Anweisungen auf Textgrundlage über eine Datenübertragungsschnittstelle auf Textgrundlage des intelligenten Tutors ausgeben, generieren hierin offenbarte Ausführungsformen dynamisch eine entsprechende Anweisung, die eine Visualisierungsschnittstelle des intelligenten Tutors modifiziert. Desgleichen, wenn der Benutzer ein grafisches Element der Visualisierungsschnittstelle modifiziert, generieren hierin offenbarte Ausführungsformen der Erfindung eine Anweisung in natürlicher Sprache, die für die Datenübertragungsschnittstelle des intelligenten Tutors bereitgestellt wird.

**[0007]** **Fig. 1** veranschaulicht beispielhafte Komponenten eines intelligenten Tutor-Systems **100**, das eine verbesserte visuelle Dialogschnittstelle bereitstellt. Im Allgemeinen ist ein intelligenter Tutor eine Anwendung, die konfiguriert ist, um eine virtuelle Lernumgebung für Benutzer bereitzustellen. Wie gezeigt, enthält der intelligente Tutor **100** eine Textschnittstelle **101**, eine Übersetzungskomponente **102** und eine Visualisierungskomponente **103**. Die Textschnittstelle **101** stellt einen Datenübertragungskanal auf Textgrundlage zwischen einem Benutzer und einer Chat-Engine **110** bereit. Im Allgemeinen ist die Chat-Engine **110** ein „Bot“, der mit dem Benutzer über Text, Emoticons, Symbole und dergleichen kommuniziert. Zum Beispiel kann die Chat-Engine **110** eine Frage mit Mehrfachauswahl an den Benutzer ausgeben. Der Benutzer kann anschließend eine Benutzereingabe **111** bereitstellen, die eine Antwort angibt. Die Chat-Engine **110** kann anschließend die Antwort validieren, die in der Benutzereingabe **111** bereitgestellt wurde, und ein Ergebnis der Validierung zurückgeben (z.B. ob die Antwort des Benutzers richtig oder falsch ist). Auf diese Weise können die Chat-Engine **110** und der Benutzer kommunizieren, wobei der Benutzer Lernanweisungen von dem intelligenten Tutor **100** empfängt.

**[0008]** Die Visualisierungskomponente **103** ist konfiguriert, um grafische Lernelemente zur Anzeige auszugeben, wie beispielsweise die beispielhaften grafischen Lernelemente **131** bis **135**. Die beispielhaften grafischen Lernelemente enthalten jeden belie-

bigen Typ eines grafischen Objekts, wie beispielsweise Abbildungen, Grafiken, schematische Darstellungen, anklickbare Konzeptknoten und dergleichen. Wie gezeigt, kann ein Benutzer mit dem grafischen Lernelementen **131** bis **135** über eine Benutzereingabe **136** interagieren. Zum Beispiel kann ein Benutzer die Farbe von einem oder mehreren der grafischen Lernelemente **131** bis **135** ändern, um sein Niveau eines Wohlfühlens mit der jedem jeweiligen grafischen Lernelement **131** bis **135** zugehörigen Thematik wiederzugeben.

**[0009]** Die Übersetzungskomponente **102** ist im Allgemeinen zum Verbessern der Textschnittstelle **101** und der Visualisierungskomponente **103** konfiguriert. Wenn die Benutzereingabe **111** in der Textschnittstelle **101** zum Beispiel angibt, „Zeigen Sie mir einen grünen Kreis“, würde die Übersetzungskomponente **102** eine Programmcodeanweisung generieren, die, wenn sie durch die Visualisierungskomponente **103** verarbeitet wird, den angeforderten grünen Kreis generiert, darstellt und ausgibt. Wenn die Benutzereingabe **136** als weiteres Beispiel eines der grafischen Lernelemente **131** bis **135** auf einen neuen Platz verschiebt, empfängt die Übersetzungskomponente **102** eine Programmcodeanweisung, die die Verschiebung beschreibt, und generiert eine Anweisung in natürlicher Sprache, die die Verschiebung wiedergibt (z.B. „Sie haben das Objekt verschoben“), die dann über die Textschnittstelle **101** ausgegeben wird.

**[0010]** Wie gezeigt, enthält die Übersetzungskomponente **102** einen Ereignis-Mapper **120**, einen oder mehrere Klassifizierer **121**, einen Entität-zu-Text-Mapper **122**, ein klassifiziertes Datenmodell **123** und einen Datenspeicher mit visuellen Befehlen **124**. Der Ereignis-Mapper **120** ordnet von der Chat-Engine **110** ausgegebenen Text Ereignissen und/oder Konzepten zu. Im Allgemeinen kann die Konversation zwischen der Chat-Engine **110** und dem Benutzer mehrere Stufen enthalten (z.B. Frage, Antwort, Suche, Vorschläge, Vergleiche usw.). Der Ereignis-Mapper **120** ordnet den Text zu derartigen Ereignissen zu. Wenn ein Benutzer zum Beispiel eine richtige Antwort auf eine Frage in der Textschnittstelle bereitstellt, würde der Ereignis-Mapper eine Antwort von der Chat-Engine **110** einem beispielhaften Konzept für „richtige Antwort“ zuordnen. Die Klassifizierer **121** sind für Klassifizierer natürlicher Sprache, Verarbeitungsalgorithmen für natürliche Sprache und Sprach-Parser repräsentativ, wie zum Beispiel ein Parser, der Benutzerimperative (oder Befehle) identifiziert. Im Allgemeinen analysieren die Klassifizierer **121** den Text der Benutzereingabe **111**, um den Text der Konversation zu klassifizieren und alle Entitäten in dem Text der Konversation zu identifizieren (z.B. Personen, Orte, Dinge, Lernkonzepte usw.). Die Klassifizierer **121** stellen die klassifizierten Daten anschließend für das klassifizierte Datenmodell **123** bereit,

das ein Maschinenlernmodell ist, das einen visuellen Befehl **124** auf Grundlage der klassifizierten Daten generiert. Die visuellen Befehle **124** sind Codeanweisungen, die durch die Visualisierungskomponente **103** ausführbar sind, um die Benutzerschnittstelle entsprechend zu modifizieren. Die Codeanweisungen können jeder beliebige Typ von Programmiersprache sein, der für visuelle Ausgaben ausgelegt ist, wie beispielsweise R (z.B. ggplot2), JavaScript, MATLAB oder irgendeine andere Sprache auf Grundlage der Grammatik von Grafik. Eine beispielhafte Codeanweisung zum Generieren eines Scatter-Plots in einer Grafik ist „ggplot(data, aes(x=my\_x, y=my\_y)) + geom\_point(shape=1)“. Ganz allgemein werden die Klassifizierer **121** daher an derartigen Beispielanweisungen geschult, um Klassifizierungen anhand von Antworten auf eine unbegrenzte Anzahl von eingegebenen Anfragen zu erzeugen.

**[0011]** Wenn die Benutzereingabe **136** von der Visualisierungskomponente **103** empfangen wird, wird ein visueller Befehl **124** (z.B. eine Codeanweisung) generiert, um die Benutzereingabe **136** zu beschreiben und zu bewirken. Wenn der Benutzer zum Beispiel die Form des Kreises **131** in ein Quadrat ändert, wird ein visueller Befehl **124** einer Codeanweisung durch die Visualisierungskomponente **103** generiert, der, wenn er ausgeführt wird, die Form des Kreises **131** in ein Quadrat ändert. Eine derartige Codeanweisung kann auch in jedem beliebigen Typ von Programmiersprache vorliegen, wie beispielsweise R (z.B. ggplot2), JavaScript, MATLAB oder irgendeine andere Sprache auf Grundlage der Grammatik von Grafik. Die visuellen Befehle **124** werden anschließend für die Übersetzungskomponente **102** bereitgestellt, die anschließend den Entität-zu-Text-Mapper **122** auf die empfangenen visuellen Befehle **124** anwendet. Der Entität-zu-Text-Mapper **122** ist ein Maschinenlernmodell, das eine Anweisung in natürlicher Sprache auf Grundlage eines empfangenen visuellen Befehls **124** generiert. Zum Beispiel kann der Entität-zu-Text-Mapper **122** die Anweisung in natürlicher Sprache „Sie haben den Kreis in ein Quadrat geändert“ in Reaktion auf die Codeanweisung generieren, die den Kreis **131** in ein Quadrat geändert hat. Der Entität-zu-Text-Mapper **122** kann anschließend die Anweisung in natürlicher Sprache für die Chat-Engine **110** bereitstellen, die die Anweisung in natürlicher Sprache über die Textschnittstelle **101** an den Benutzer ausgibt.

**[0012]** Fig. 2A stellt eine beispielhafte grafische Benutzeroberfläche (GUI) **200** dar, die verbesserte visuelle Dialoge enthält, die durch den intelligenten Tutor **100** gemäß einer Ausführungsform bereitgestellt werden. Wie gezeigt, enthält die GUI **200** eine Chat-Schnittstelle **201** und eine Visualisierungsschnittstelle **202**. Die Chat-Schnittstelle **201**, eine beispielhafte Ausgabe der Textschnittstelle **101**, ist eine GUI, die es einem Benutzer und der Chat-Engine **110** ermög-

licht, über Text und andere Symbole miteinander zu kommunizieren. Die Visualisierungsschnittstelle **202** ist eine GUI, die durch die Visualisierungskomponente **103** generiert wird, die Grafikobjekte ausgibt, die verwendet werden, um Benutzern das Lernen zu erleichtern.

**[0013]** Wie gezeigt, hat die Chat-Engine **110** eine Frage mit Mehrfachauswahl für einen beispielhaften Benutzer „student1“ über die Chat-Schnittstelle **201** dargestellt. Die Frage mit Mehrfachauswahl bezieht sich auf das Langzeitgedächtnis. Wie des Weiteren gezeigt, stellt die Visualisierungsschnittstelle **202** für den Benutzer eine Mehrzahl von Konzeptknoten **203** bis **207** dar, die sich auf das menschliche Gedächtnis beziehen. Der Benutzer kann auf einen der Konzeptknoten **203** bis **207** klicken, um zu jedem Konzept zugehörige Informationen zu erhalten, wie zum Beispiel Lernprogramme, Abbildungen und dergleichen. Wie gezeigt, hat der Benutzer „student1“ eine Antwort auf die Frage mit Mehrfachauswahl über die Chat-Schnittstelle **201** weitergeleitet.

**[0014]** Fig. 2B stellt eine beispielhafte Ausgabe der GUI **200** dar, nachdem der intelligente Tutor **100** bestimmt hat, dass die Mehrfachauswahl-Antwort, die der Benutzer „student1“ gegeben hat, falsch ist. Wie gezeigt, enthält die Chat-Schnittstelle **201** eine Textantwort von der Chat-Engine **110**, die angibt, dass die Antwort falsch ist. Wie gezeigt, wurde die GUI **200** jedoch aktualisiert, um den Konzeptknoten **205** optisch hervorzuheben, da dieser Konzeptknoten dem Konzept von „Langzeitgedächtnis“ entspricht, das ein Konversationsthema in der Chat-Schnittstelle **201** ist. Wie vorher angegeben, wird der Text der Chat-Schnittstelle **201** durch die Klassifizierer **121** analysiert, was Klassifizierer natürlicher Sprache, Prozessoren für natürliche Sprache und andere Parser-Typen umfasst. Desgleichen stellt die Chat-Engine **110** Text der Konversation für den Ereignis-Mapper **120** bereit, der den Text einem entsprechenden Ereignis zuordnet. In dem in Fig. 2B dargestellten Beispiel kann der Ereignis-Mapper **120** den Text der Konversation dem Ereignis „falsche Antwort“ zuordnen, da der Benutzer auf die Frage mit Mehrfachauswahl eine falsche Antwort gegeben hat. Desgleichen können die Klassifizierer **121** die von dem Benutzer bereitgestellte Eingabe klassifizieren, um zu bestimmen, dass der Benutzer das Konzept „Langzeitgedächtnis“ besser verstehen muss. Daher kann das klassifizierte Datenmodell **123** eine Codeanweisung (z.B. einen visuellen Befehl **124**) auf Grundlage des Texts der Chat-Schnittstelle **201** generieren, der angibt, den Konzeptknoten **205** optisch hervorzuheben. Die Codeanweisung hebt bei Ausführung durch die Visualisierungskomponente **103** den Konzeptknoten **105** in der GUI **202** optisch hervor.

**[0015]** Fig. 2C stellt eine Ausführungsform der Erfindung dar, in der der Benutzer „student1“ die GUI

modifiziert hat. Insbesondere hat der Benutzer „student1“ die Farbe (oder Schattierung) des Konzeptknotens **205** geändert, was z.B. wiedergibt, dass der Benutzer die entsprechenden Lernprogramme gelesen hat und mit dem Konzept von Langzeitgedächtnis besser vertraut ist. Um die Änderung zu bewirken, generiert die Visualisierungskomponente **103** eine Codeanweisung, die das Erscheinungsbild des Konzeptknotens **205** modifiziert, und führt sie aus. Die Visualisierungskomponente **103** stellt die Codeanweisung für die Übersetzungskomponente **102** bereit. Die visuellen Befehle **124** stellen dann die Codeanweisung für den Entität-zu-Text-Mapper **122** bereit, die eine Anweisung in natürlicher Sprache generiert, die der Codeanweisung entspricht. Die Anweisung in natürlicher Sprache wird dann in der Chat-Schnittstelle **201** ausgegeben. Wie gezeigt, stellt die Chat-Schnittstelle **201** daher die Chat-Engine **110** so dar, dass sie die beispielhafte Anweisung in natürlicher Sprache von „Benutzer student1 hat die Farbe des Langzeitgedächtnis-Konzeptknotens geändert“ ausgibt. Während ein Ändern der Farbe des Konzeptknotens **205** ein Beispiel für eine Benutzereingabe ist, die die GUI **202** verändert, wird jeder beliebige Typ einer Benutzerinteraktion auf ähnliche Weise verarbeitet. Wenn der Benutzer zum Beispiel den Konzeptknoten **203** auf einen anderen Platz verschoben hat, kann die Übersetzungskomponente **102** eine beispielhafte Anweisung in natürlicher Sprache „Benutzer student1 hat den deklarativen Erinnerung zugehörigen Konzeptknoten verschoben“ generieren.

**[0016]** Fig. 2D stellt eine Ausführungsform der Erfindung dar, in der der Benutzer eine imperative Anweisung über die GUI **201** an die Chat-Engine **110** weiterleitet. Insbesondere, wie gezeigt, hat der Benutzer angegeben „Bitte Informationen zum Kurzzeitgedächtnis anzeigen“. Die Komponenten der Übersetzungskomponente **102** analysieren die Anweisung und bestimmen, dass der Benutzer die Chat-Engine **100** angewiesen hat, eine Aktion auszuführen. In Reaktion darauf generiert die Übersetzungskomponente **102** eine Codeanweisung, die, wenn sie durch die Visualisierungskomponente **103** ausgeführt wird, den Konzeptknoten **204** optisch hervorhebt, der dem Kurzzeitgedächtnis entspricht, wie in der GUI **202** wiedergegeben.

**[0017]** Zusätzlich zur optischen Hervorhebung des Konzeptknotens **204** kann die Übersetzungskomponente **102** Anweisungen generieren, die für Benutzer Tipps über die GUI **202** bereitstellen. Des Weiteren kann die Übersetzungskomponente **102** das Layout der GUIs **201**, **202** (z.B. nebeneinander, die GUI **201** in die GUI **202** eingebettet usw.) modifizieren. Außerdem kann die Übersetzungskomponente **102** Anweisungen generieren, die Pfade von einer visuellen Entität in der GUI **202** zu einer anderen optisch hervorheben, wie zum Beispiel von einem Konzeptknoten, der einem Thema entspricht, das der Benut-

zer versteht, zu einem Antwortkonzept in der Grafik. Des Weiteren kann die Übersetzungskomponente **102** Anweisungen generieren, die mehr als eine Entität in der GUI **202** optisch hervorheben, wie zum Beispiel ein Hervorheben der Konzeptknoten **204** bis **205**, um dem Benutzer die Unterschiede zwischen Kurz- und Langzeitgedächtnis aufzuzeigen. Des Weiteren kann der intelligente Tutor **100** Konzeptknoten generieren, die es einem Benutzer erlauben, große Mengen von Text zu lesen/aufzunehmen, oder Vergleiche von Kombinationen von Elementen auf einer langen Elementliste anzustellen, wenn derartige Anzahlen von Kombinationen für eine Vorabschulung unpraktisch groß wären. Neben dem Lesen des Texts führt der intelligente Tutor **100** einen visuellen Leitfaden zu den Wissensentitäten ein, die in dem Text enthalten sind, indem Konzepte in dem Text identifiziert werden und eine Grafik der Konzeptknoten zur Anzeige in der GUI **202** erstellt wird. Die Grafik der Konzeptknoten ermöglicht es den Benutzern, mehr über jedes zugehörige Thema durch Auswahl eines bestimmten Knotens zu erfahren.

**[0018]** Fig. 3 ist ein Ablaufplan, der ein beispielhaftes Verfahren **300** zum Bereitstellen eines verbesserten visuellen Dialogsystems für intelligente Tutoren veranschaulicht. Wie gezeigt, beginnt das Verfahren **300** am Block **310**, in dem die Klassifizierer **121** geschult werden, der unter Bezugnahme auf Fig. 4 ausführlicher beschrieben wird. Im Allgemeinen wird zum Schulen eines Klassifizierers eine gekennzeichnete Eingabe bereitgestellt, die dem Klassifizierer ein Berechnen eines gewünschten Results ermöglicht. Zum Beispiel kann die Schulung verschiedene Anweisungen in natürlicher Sprache bereitstellen, die ähnliche Konzepte haben (z.B. „Zeigen Sie mir den grünen Kreis“, „Wo ist der grüne Kreis“, „Sehen wir uns den grünen Kreis an“ usw.). Als weiteres Beispiel kann die Schulung verschiedene Programmcodeanweisungen bereitstellen (z.B. Anweisungen, die die Visualisierungsschnittstelle **202** modifizieren), die zum Generieren von entsprechenden Anweisungen in natürlicher Sprache gekennzeichnet sind. Die Schulung der Klassifizierer **112** kann ein oder mehrere Maschinenlern- (ML) Modelle, die verwendet werden, um natürliche Sprache zu Codeanweisungen zuzuordnen, um die Elemente der Visualisierungsschnittstelle **202** zu modifizieren, und ML-Modelle erzeugen, die die generierten Codeanweisungen auf Grundlage von Benutzerinteraktionen mit der Visualisierungsschnittstelle **202** zu Anweisungen in natürlicher Sprache zuordnen.

**[0019]** Am Block **320** kann die Chat-Engine **110** über die Chat-Schnittstelle **201** eine Textkommunikation an den Benutzer ausgeben. Die Textkommunikation kann jeder beliebige Typ von Kommunikation auf Lerngrundlage sein, wie beispielsweise Fragen, Lernmaterial und dergleichen. Am Block **330** empfängt der intelligente Tutor **100** Benutzereingaben.

Die Benutzereingabe kann über die Chat-Schnittstelle **201** und/oder die Visualisierungsschnittstelle **202** empfangen werden. Zum Beispiel kann der Benutzer eine Antwort über die Chat-Schnittstelle **201** eintippen und/oder ein grafisches Objekt bearbeiten, das über die Visualisierungsschnittstelle **202** zur Anzeige ausgegeben wird. Am Block **340**, nachdem bestimmt wurde, dass die Eingabe eine Eingabe auf Textgrundlage ist, die über die Chat-Schnittstelle empfangen wurde, übersetzt die Übersetzungskomponente **102** die Texteingabe in eine Codeanweisung, um die Visualisierungsschnittstelle **201** zu modifizieren. Wenn der Benutzer zum Beispiel eine Nachricht eintippt, die feststellt „Ich verstehe Patente nicht“, kann die Übersetzungskomponente **102** eine Codeanweisung generieren, die durch die Visualisierungsschnittstelle **201** ausführbar ist, um ein Lernprogramm zu Patenten auszugeben.

**[0020]** Nachdem bestimmt worden ist, dass die Eingabe über die Visualisierungsschnittstelle **202** empfangen wurde, generiert die Übersetzungskomponente **102** am Block **350** eine Anweisung in natürlicher Sprache auf Grundlage der Codeanweisung, die auf Grundlage der Benutzereingabe generiert wurde. Wenn die Benutzereingabe zum Beispiel angibt, das Lernprogramm über Patente zu öffnen, generiert die Visualisierungsschnittstelle **202** eine Codeanweisung, die das Lernprogramm über Patente in der Visualisierungsschnittstelle **202** ausgibt. Die Codeanweisung wird dann durch die Übersetzungskomponente **102** in eine Anweisung in natürlicher Sprache übersetzt, wie zum Beispiel „Sie haben das Lernprogramm über Patente geöffnet“. Am Block **360** wird die GUI **200** auf Grundlage der übersetzten Eingabe modifiziert. Zum Beispiel kann die Anweisung in natürlicher Sprache „Sie haben das Lernprogramm über Patente geöffnet“ über die Chat-Schnittstelle **101** ausgegeben werden. Als weiteres Beispiel kann die Visualisierungskomponente **103** die Codeanweisung, die durch die Übersetzungskomponente **102** generiert wurde, zum Modifizieren der Visualisierungsschnittstelle **202** ausführen (z.B. um das Lernprogramm zu Patenten auszugeben). Am Block **370** kann die Übersetzungskomponente **102** ein Maschinenlernmodell für den aktuellen Benutzer aktualisieren. Das Benutzer-ML-Modell berücksichtigt die Eingabe, die durch den Benutzer bereitgestellt wird, alle historischen Aktionen, die eine Interaktionshistorie des Benutzers wiedergeben, und den Lernfortschritt des Benutzers. Das ML-Modell für den Benutzer wird dann von der Übersetzungskomponente **102** verwendet, um anschließend die Visualisierungsschnittstelle **202** und/oder die Chat-Schnittstelle **201** zu modifizieren.

**[0021]** Fig. 4 ist ein Ablaufplan, der ein beispielhaftes Verfahren **400** veranschaulicht, das dem Block **310** zum Schulen von Klassifizierern entspricht. Wie gezeigt, beginnt das Verfahren am Block **410**, wo-

bei die Übersetzungskomponente **410** Schulungsdatensets empfängt, die optional Benutzerhistorien und/oder Benutzerdaten enthalten können. Am Block **420** werden die Schulungsdatensets verwendet, um die Klassifizierer **121** zu schulen. Im Allgemeinen enthalten die Schulungsdatensets gekennzeichnete Eingabedaten, die eine Angabe einer „richtigen“ Antwort enthalten. Zum Beispiel können die Schulungsdatensets eine Mehrzahl von verschiedenen Befehlen und eine entsprechende Angabe des Typs jedes Benutzerbefehls enthalten. Die Schulungsdatensets können auch eine Angabe einer zu generierenden entsprechenden Codeanweisung enthalten, um die Visualisierungsschnittstelle **103** auf Grundlage des Typs jedes Benutzerbefehls zu modifizieren. Desgleichen können die Schulungsdaten Codeanweisungen enthalten, die auf Grundlage von Benutzerinteraktionen mit den grafischen Elementen der Visualisierungsschnittstelle **103**, eines der Interaktion zugehörigen Ereignisses und einer Anweisung in natürlicher Sprache generiert wurden, die dem Ereignis und/oder der Interaktion zugehörig ist. Auf Grundlage der Schulungsdaten generieren die Klassifizierer **121** Text-zu-Visualisierung-ML-Modelle am Block **430**. Am Block **440** generieren die Klassifizierer **121** Visualisierung-zu-Text-ML-Modelle.

**[0022]** Fig. 5 ist ein Ablaufplan, der gemäß einer Ausführungsform ein beispielhaftes Verfahren **500**, das dem Block **340** zum Übersetzen einer Texteingabe entspricht, zum Generieren einer Anweisung veranschaulicht, um eine Visualisierung zu modifizieren. Wie gezeigt, beginnt das Verfahren am Block **510**, an dem die Klassifizierer **121** die empfangene Texteingabe verarbeiten, um die Konzepte darin zu identifizieren. Am Block **520** ordnet die Übersetzungskomponente **102** die Konzepte einer visuellen Entsprechung zu, z.B. auf Grundlage der Text-zu-Visualisierung-ML-Modelle, die am Block **430** generiert wurden. Am Block **530** kann die Übersetzungskomponente **102** eine Programmcodeanweisung auf Grundlage der Zuordnungen generieren, die am Block **520** bestimmt wurden. Am Block **540** überträgt die Übersetzungskomponente **102** die generierte Codeanweisung zu der Visualisierungskomponente **103**. Am Block **550** führt die Visualisierungskomponente **103** die empfangene Codeanweisung aus, um die Visualisierungsschnittstelle **202** zu modifizieren.

**[0023]** Fig. 6 ist ein Ablaufplan, der ein beispielhaftes Verfahren **600** veranschaulicht, das Block **350** zum Übersetzen einer Benutzereingabe entspricht, die an einer Visualisierungsschnittstelle empfangen wird, um eine Anweisung in natürlicher Sprache zu erstellen. Wie gezeigt, beginnt das Verfahren **600** am Block **610**, an dem die Übersetzungskomponente **102** eine Programmcodeanweisung von der Visualisierungskomponente **103** empfängt, wobei die Programmcodeanweisung die Benutzereingabe beschreibt, die über die Visualisierungsschnitt-

stelle **202** empfangen wurde. Am Block **620** verwendet die Übersetzungskomponente **102** ein ML-Modell zum Klassifizieren der Programmcodeanweisung. Am Block **630** übersetzt die Übersetzungskomponente **102** die klassifizierte Programmcodeanweisung in eine Anweisung in natürlicher Sprache. Am Block **640** gibt die Chat-Engine **110** über die Chat-Schnittstelle **201** die Anweisung in natürlicher Sprache aus.

**[0024]** Fig. 7 ist ein Blockschaubild, das ein beispielhaftes System **700** veranschaulicht, das ein verbessertes visuelles Dialogsystem für intelligente Tutoren bereitstellt. Das vernetzte System **700** enthält einen Computer **702**. Der Computer **702** kann über ein Netzwerk **730** auch mit anderen Computern (z.B. einem Client-System **750**) verbunden werden. Im Allgemeinen kann das Netzwerk **730** ein Telekommunikationsnetzwerk und/oder ein Weitverkehrsnetzwerk (WAN) sein. In einer besonderen Ausführungsform ist das Netzwerk **730** das Internet.

**[0025]** Der Computer **702** enthält im Allgemeinen einen Prozessor **704**, der Anweisungen und Daten über einen Bus **720** aus einem Arbeitsspeicher **706** und/oder einem Speicher **708** erhält. Der Computer **702** kann auch eine oder mehrere Netzwerk-Schnittstelleneinheiten **718**, Eingabeeinheiten **722** und Ausgabeeinheiten **724** enthalten, die mit dem Bus **720** verbunden sind. Der Computer **702** steht allgemein unter der Steuerung eines Betriebssystems (nicht gezeigt). Beispiele für Betriebssysteme enthalten das UNIX-Betriebssystem, Versionen des Windows-Betriebssystems von Microsoft und Distributionen des Linux-Betriebssystems. (UNIX ist eine eingetragene Marke von The Open Group in den Vereinigten Staaten und anderen Ländern. Microsoft und Windows sind Marken der Microsoft Corporation in den USA, anderen Ländern oder beidem. Linux ist eine eingetragene Marke von Linus Torvalds in den Vereinigten Staaten, anderen Ländern oder beidem.) Ganz allgemein können alle Betriebssysteme verwendet werden, die die hierin offenbarten Funktionen unterstützen. Der Prozessor **704** ist eine programmierbare Logikeinheit, die Anweisungs-, Logik- und mathematische Verarbeitung ausführt und für eine oder mehrere CPUs stehen kann. Die Netzwerk-Schnittstelleneinheit **718** kann jeder Typ von Netzwerk-Kommunikationseinheit sein, die es dem Computer **702** ermöglicht, mit anderen Computern über das Netzwerk **730** Daten auszutauschen.

**[0026]** Der Speicher **708** steht für Festplattenlaufwerke, Solid-State-Drives, Flashspeichereinheiten, optische Medien und dergleichen. Im Allgemeinen speichert der Speicher **708** Anwendungsprogramme und Daten zur Verwendung durch den Computer **702**. Außerdem kann in Betracht gezogen werden, dass der Arbeitsspeicher **706** und der Speicher **708** einen physisch außerhalb befindlichen Speicher enthalten,

zum Beispiel auf einem anderen Computer, der über den Bus **720** mit dem Computer **702** verbunden ist.

**[0027]** Die Eingabeeinheit **722** kann jede Einheit zum Bereitstellen von Eingaben in den Computer **702** sein. Zum Beispiel kann eine Tastatur und/oder eine Maus verwendet werden. Die Eingabeeinheit **722** stellt eine breite Vielfalt von Eingabeeinheiten dar, darunter Tastaturen, Mäuse, Controller usw. Ferner kann die Eingabeeinheit **722** eine Gruppe von Schaltflächen, Schaltern oder anderen Mechanismen physischer Einheiten zum Steuern des Computers **702** enthalten. Die Ausgabeeinheit **724** kann Ausgabeeinheiten wie zum Beispiel Monitore, Berührungsbildschirmanzeigen usw. enthalten.

**[0028]** Wie gezeigt, enthält der Arbeitsspeicher **706** den intelligenten Tutor **101**, der oben ausführlicher beschrieben wurde. Wie gezeigt, enthält der Speicher **708** die ML-Modelle **715** und speichert alle der oben beschriebenen ML-Modelle. Im Allgemeinen ist das System **700** konfiguriert, um die gesamte oben unter Bezugnahme auf die Fig. 1 bis Fig. 6 beschriebene Funktionalität umzusetzen. Das Client-System **750** ist ein Benutzersystem, das eine Benutzer-Instanz des intelligenten Tutors **100** enthält. Der Benutzer greift auf die Funktionalität des intelligenten Tutors **100** über mindestens die GUIs **200** bis **202** zu, die auf einer Anzeigeeinheit (nicht dargestellt) des Client-Systems **750** angegeben sind.

**[0029]** Vorteilhafterweise stellen die hierin offenbarten Ausführungsformen der Erfindung Techniken bereit, um die verschiedenen Komponenten eines intelligenten Tutor-Systems stärker zu integrieren. Insbesondere übersetzen die hierin offenbarten Ausführungsformen der Erfindung Benutzertext und Text, der von einem Chat-Bot in einer Chat-Schnittstelle generiert wurde, in Programmcodeanweisungen, die eine Visualisierungsschnittstelle des intelligenten Tutors modifizieren. Außerdem generieren die hierin offenbarten Ausführungsformen der Erfindung Anweisungen in natürlicher Sprache auf Grundlage der Codeanweisungen, die in Reaktion auf eine Benutzereingabe generiert werden, die die Visualisierungsschnittstelle modifiziert. Dadurch wird die Funktionalität des intelligenten Tutor-Systems verbessert und stellt für den Benutzer eine bessere Lernerfahrung bereit. Dadurch wird auch ein Modell einer Benutzerinteraktion erzeugt, das visuell-räumliche Elemente sowie natürliche Sprache enthält, wodurch eine Datenaufnahme und Schulung von Maschinenlernmodellen in nicht räumlichen Domänen ermöglicht wird.

**[0030]** Die Beschreibungen der verschiedenen Ausführungsformen der vorliegenden Erfindung wurden zum Zweck der Veranschaulichung erstellt, sie sollen aber keineswegs erschöpfend oder auf die offenbarten Ausführungsformen eingeschränkt sein. Für Fachleute sind viele Modifizierungen und Variationen

offenkundig, die nicht von dem Schutzzumfang der Erfindung abweichen. Die hierin verwendete Terminologie wurde gewählt, um die Grundgedanken der Erfindung, der praktischen Anwendung oder technischen Verbesserung gegenüber auf dem Markt gefundenen Technologien bestmöglich zu erklären oder anderen Fachleuten das Verständnis der hierin offenbarten Ausführungsformen zu ermöglichen.

**[0031]** Im Vorgenannten wird auf die hierin vorgestellten Ausführungsformen der Erfindung Bezug genommen. Der Schutzzumfang der vorliegenden Erfindung ist jedoch nicht auf spezifische beschriebene Ausführungsformen beschränkt. Stattdessen wird jede Kombination der genannten Merkmale und Elemente, gleichgültig, ob sie zu verschiedenen Ausführungsformen in Beziehung stehen, zum Umsetzen und Ausüben der Erfindung in Erwägung gezogen. Obwohl hierin offenbarte Ausführungsformen der Erfindung des Weiteren Vorteile gegenüber anderen möglichen Lösungen oder gegenüber dem Stand der Technik erlangen können, stellt das Erreichen oder Nicht-Erreichen eines bestimmten Vorteils durch eine bestimmte Ausführungsform keine Einschränkung des Schutzzumfangs der vorliegenden Offenbarung dar. Daher sind die genannten Aspekte, Merkmale, Ausführungsformen und Vorteile nur veranschaulichend und sind nicht als Elemente oder Einschränkungen der Ansprüche im Anhang zu betrachten, es sei denn in Fällen, in denen sie explizit in einem Anspruch (bzw. mehreren Ansprüchen) zitiert werden. Desgleichen soll die Bezugnahme auf „die Erfindung“ nicht als Verallgemeinerung irgendeines hierin offenbarten Erfindungsgegenstands ausgelegt werden und soll nicht als ein Element oder eine Einschränkung der Ansprüche im Anhang betrachtet werden, es sei denn in Fällen, in denen sie explizit in einem Anspruch (bzw. mehreren Ansprüchen) zitiert werden.

**[0032]** Aspekte der Erfindung können in der Form einer vollständigen Hardware-Ausführungsform, einer vollständigen Software-Ausführungsform (einschließlich Firmware, residente Software, Mikrocode usw.) oder einer Ausführungsform vorliegen, die Software- und Hardware-Aspekte kombiniert, auf die alle hierin allgemein als „Schaltung“, „Modul“ oder „System“ Bezug genommen werden kann.

**[0033]** Die Erfindung kann in einem System, einem Verfahren und/oder einem Computerprogrammprodukt verkörpert sein. Das Computerprogrammprodukt kann ein durch einen Computer lesbares Speichermedium (oder -medien) enthalten, auf dem durch einen Computer lesbare Programmanweisungen gespeichert sind, um einen Prozessor dazu zu veranlassen, Aspekte der Erfindung auszuführen.

**[0034]** Bei dem durch einen Computer lesbaren Speichermedium kann es sich um eine physische

Einheit handeln, die Anweisungen zur Verwendung durch eine Einheit zur Ausführung von Anweisungen beibehalten und speichern kann. Das durch einen Computer lesbare Speichermedium kann zum Beispiel eine elektronische Speichereinheit, eine magnetische Speichereinheit, eine optische Speichereinheit, eine elektromagnetische Speichereinheit, eine Halbleiter-Speichereinheit oder jede geeignete Kombination aus dem Vorgenannten sein, es ist aber nicht darauf beschränkt. Zu einem durch einen Computer lesbaren Speichermedium können eine elektronische Speichereinheit, eine Magnetspeichereinheit, eine optische Speichereinheit, eine elektromagnetische Speichereinheit, eine Halbleiterspeichereinheit, oder jede geeignete Kombination des Vorgenannten gehören, es ist aber nicht darauf beschränkt. Zu einer nicht erschöpfenden Liste spezifischerer Beispiele des durch einen Computer lesbaren Speichermediums gehören die Folgenden: eine tragbare Computerdiskette, eine Festplatte, ein Direktzugriffsspeicher (RAM), ein Nur-Lese-Speicher (ROM), ein löschbarer programmierbarer Nur-Lese-Speicher (EPROM bzw. Flashspeicher), ein statischer Direktzugriffsspeicher (SRAM), ein tragbarer CD-ROM, eine DVD, ein Speicherstick, eine Diskette, eine mechanisch codierte Einheit wie zum Beispiel Lochkarten oder erhabene Strukturen in einer Rille, auf denen Anweisungen gespeichert sind, und jede geeignete Kombination des Vorgenannten. Ein durch einen Computer lesbares Speichermedium soll, wie hierin verwendet, nicht als flüchtige Signale an sich aufgefasst werden, wie zum Beispiel Funkwellen oder andere sich frei ausbreitende elektromagnetische Wellen, elektromagnetische Wellen, die sich durch einen Wellenleiter oder andere Übertragungsmedien ausbreiten (z.B. durch einen Lichtwellenleiter geleitete Lichtimpulse) oder durch einen Draht übertragene elektrische Signale.

**[0035]** Hierin beschriebene durch einen Computer lesbare Programmanweisungen können von einem durch einen Computer lesbaren Speichermedium auf jeweilige Datenverarbeitungs-/Verarbeitungseinheiten oder über ein Netzwerk wie zum Beispiel das Internet, ein lokales Netzwerk, ein Weitverkehrsnetzwerk und/oder ein drahtloses Netzwerk auf einen externen Computer oder eine externe Speichereinheit heruntergeladen werden. Das Netzwerk kann Kupferübertragungskabel, Lichtwellenübertragungsleiter, drahtlose Übertragung, Leitwegrechner, Firewalls, Vermittlungseinheiten, Gateway-Computer und/oder Edge-Server aufweisen. Eine Netzwerkadapterkarte oder Netzwerkschnittstelle in jeder Datenverarbeitungs-/Verarbeitungseinheit empfängt durch einen Computer lesbare Programmanweisungen aus dem Netzwerk und leitet die durch einen Computer lesbaren Programmanweisungen zur Speicherung in einem durch einen Computer lesbaren Speichermedium innerhalb der entsprechenden Datenverarbeitungs-/Verarbeitungseinheit weiter.



**[0036]** Bei durch einen Computer lesbaren Programmanweisungen zum Ausführen von Operationen der vorliegenden Offenbarung kann es sich um Assembler-Anweisungen, ISA-Anweisungen (Instruction-Set-Architecture), Maschinenanweisungen, maschinenabhängige Anweisungen, Mikrocode, Firmware-Anweisungen, zustandssetzende Daten oder entweder Quellcode oder Objektcode handeln, die in einer beliebigen Kombination aus einer oder mehreren Programmiersprachen geschrieben sind, darunter objektorientierte Programmiersprachen wie Smalltalk, C++ oder dergleichen, sowie herkömmliche prozedurale Programmiersprachen wie die Programmiersprache „C“ oder ähnliche Programmiersprachen. Die durch einen Computer lesbaren Programmanweisungen können vollständig auf dem Computer des Benutzers, teilweise auf dem Computer des Benutzers, als eigenständiges Software-Paket, teilweise auf dem Computer des Benutzers und teilweise auf einem entfernt angeordneten Computer oder vollständig auf dem entfernt angeordneten Computer oder Server ausgeführt werden. In dem letzteren Szenario kann der entfernt angeordnete Computer mit dem Computer des Benutzers durch jeden Typ von Netzwerk verbunden werden, darunter ein lokales Netzwerk (LAN) oder ein Weitverkehrsnetzwerk (WAN), oder die Verbindung kann mit einem externen Computer hergestellt werden (zum Beispiel über das Internet unter Verwendung eines Internet-Diensteanbieters). In einigen Ausführungsformen können elektronische Schaltungen, darunter zum Beispiel programmierbare Logikschaltungen, feldprogrammierbare Gatter-Anordnungen (FPGA, field programmable gate arrays) oder programmierbare Logikanordnungen (PLA, programmable logic arrays) die computerlesbaren Programmanweisungen ausführen, indem sie Zustandsinformationen der computerlesbaren Programmanweisungen nutzen, um die elektronischen Schaltungen zu personalisieren, um Aspekte der vorliegenden Offenbarung durchzuführen.

**[0037]** Aspekte der vorliegenden Offenbarung werden hierin unter Bezugnahme auf Veranschaulichungen von Ablaufplänen und/oder Blockschaubildern von Verfahren, Vorrichtungen (Systemen) und Computerprogrammprodukten gemäß Ausführungsformen der Erfindung beschrieben. Es sollte klar sein, dass jeder Block der Ablaufplanveranschaulichungen und/oder der Blockschaubilder und Kombinationen von Blöcken in den Ablaufplanveranschaulichungen und/oder den Blockschaubildern mittels durch einen Computer lesbare Programmanweisungen umgesetzt werden können.

**[0038]** Diese durch einen Computer lesbaren Programmanweisungen können einem Prozessor eines Universalcomputers, eines Spezialcomputers oder einer anderen programmierbaren Datenverarbeitungsvorrichtung bereitgestellt werden, um eine

Maschine zu erzeugen, sodass die über den Prozessor des Computers bzw. eine andere programmierbare Datenverarbeitungsvorrichtung ausgeführten Anweisungen ein Mittel zur Umsetzung der in dem Block bzw. den Blöcken der Ablaufpläne und/oder der Blockschaubilder angegebenen Funktionen/Schritte erzeugen. Diese durch einen Computer lesbaren Programmanweisungen können auch auf einem durch einen Computer lesbaren Speichermedium gespeichert sein, das einen Computer, eine programmierbare Datenverarbeitungsvorrichtung und/oder andere Einheiten so steuern kann, dass sie auf eine bestimmte Art funktionieren, sodass das durch einen Computer lesbare Speichermedium, auf dem Anweisungen gespeichert sind, ein Herstellungsprodukt mit Anweisungen darauf aufweist, die Aspekte der/des in dem Block bzw. den Blöcken des Ablaufplans und/oder der Blockschaubilder angegebenen Funktion/Schritts umsetzen.

**[0039]** Die durch einen Computer lesbaren Programmanweisungen können auch auf einen Computer, eine andere programmierbare Datenverarbeitungsvorrichtung oder eine andere Einheit geladen werden, um das Ausführen einer Reihe von Operationen auf dem Computer bzw. der anderen programmierbaren Vorrichtung oder anderen Einheit zu veranlassen, um einen durch einen Computer umgesetzten Prozess zu erzeugen, sodass die auf dem Computer, einer anderen programmierbaren Vorrichtung oder einer anderen Einheit ausgeführten Anweisungen die in dem Block bzw. den Blöcken der Ablaufpläne und/oder der Blockschaubilder angegebenen Funktionen/Schritte umsetzen.

**[0040]** Der Ablaufplan und die Blockschaubilder in den Figuren veranschaulichen die Architektur, die Funktionalität und den Betrieb von möglichen Umsetzungen von Systemen, Verfahren und Computerprogrammprodukten gemäß verschiedenen Ausführungsformen der vorliegenden Erfindung. In diesem Zusammenhang kann jeder Block in den Ablaufplänen oder den Blockschaubildern ein Modul, ein Segment oder einen Teil von Anweisungen darstellen, die eine oder mehrere ausführbare Anweisungen zum Umsetzen der bestimmten logischen Funktion (en) aufweisen. In einigen alternativen Umsetzungen können die in dem Block angegebenen Funktionen in einer anderen Reihenfolge als in den Figuren gezeigt auftreten. Zum Beispiel können zwei nacheinander gezeigte Blöcke tatsächlich im Wesentlichen parallel ausgeführt werden, oder die Blöcke können manchmal in der umgekehrten Reihenfolge ausgeführt werden, was von der beteiligten Funktionalität abhängt. Es ist ferner anzumerken, dass jeder Block der Blockschaubilder und/oder der Ablaufplanveranschaulichungen sowie Kombinationen von Blöcken in den Blockschaubildern und/oder der Ablaufplanveranschaulichung durch spezielle auf Hardware beruhende Systeme umgesetzt werden können, wel-

che die angegebenen Funktionen oder Handlungen durchführen oder Kombinationen aus Spezial-Hardware und Computeranweisungen ausführen.

**[0041]** Ausführungsformen der Offenbarung können für Endbenutzer über eine Cloud-Computing-Infrastruktur bereitgestellt werden. Cloud Computing (Cloud-Datenverarbeitung) betrifft im Allgemeinen die Bereitstellung von skalierbaren Datenverarbeitungsressourcen als einen Dienst über ein Netzwerk. Formaler ausgedrückt kann Cloud Computing als eine Datenverarbeitungsfunktion definiert werden, die eine Abstraktion zwischen der Datenverarbeitungsressource und ihrer zugrunde liegenden technischen Architektur bereitstellt (z.B. Server, Speicher, Netzwerke), die einen bequemen On-Demand-Netzwerkzugriff (Netzwerkzugriff bei Bedarf) auf einen gemeinsam genutzten Pool von konfigurierbaren Datenverarbeitungsressourcen ermöglichen, die mit minimalem Verwaltungsaufwand oder Interaktion mit einem Dienstanbieter rasch bereitgestellt und freigegeben werden können. Daher gestattet Cloud Computing einem Benutzer das Zugreifen auf virtuelle Datenverarbeitungsressourcen (z.B. Speicher, Daten, Anwendungen und sogar vollständig virtualisierte Datenverarbeitungssysteme) in der „Cloud“ ohne Berücksichtigung der zugrunde liegenden physischen Systeme (oder Standorte dieser Systeme), die zum Bereitstellen der Datenverarbeitungsressourcen verwendet werden.

**[0042]** Typischerweise werden Cloud-Computing-Ressourcen einem Benutzer auf der Grundlage einer nutzungsabhängigen Bezahlung bereitgestellt, bei der Benutzern nur die tatsächliche Nutzung der Datenverarbeitungsressourcen berechnet wird (z.B. eine von einem Benutzer in Anspruch genommene Speicherplatzmenge oder eine Anzahl von virtualisierten Systemen, die durch den Benutzer instanziiert werden). Ein Benutzer kann jederzeit und von überall im Internet auf jede der Ressourcen zugreifen, die sich in der Cloud befinden. Im Kontext der Erfindung kann ein Benutzer auf Anwendungen oder dazugehörige Daten zugreifen, die in der Cloud verfügbar sind. Zum Beispiel könnte der intelligente Tutor **100** auf einem Datenverarbeitungssystem in der Cloud ausgeführt werden. In einem derartigen Fall kann der intelligente Tutor **100** die ML-Modelle **715** an einem Speicherplatz in der Cloud generieren und speichern. Dies gestattet einem Benutzer, auf diese Informationen von jedem Datenverarbeitungssystem aus zuzugreifen, das an ein mit der Cloud verbundenes Netzwerk angeschlossen ist (z.B. das Internet).

**[0043]** Zwar bezieht sich das Vorhergehende auf Ausführungsformen der Erfindung, doch sind andere und weitere Ausführungsformen der Erfindung denkbar, ohne von deren grundlegendem Schutzzumfang

abzuweichen, und deren Schutzzumfang wird durch die folgenden Ansprüche festgelegt.

### Patentansprüche

1. Verfahren, aufweisend:  
ein Empfangen einer Texteingabe über eine Chat-Schnittstelle einer Tutor-Anwendung;  
ein Identifizieren eines Konzepts der Texteingabe durch mindestens einen Klassifizierer, der auf die Texteingabe angewendet wird;  
ein Zuordnen des Konzepts in der Texteingabe zu mindestens einer visuellen Aktion oder einem ersten visuellen Objekt;  
ein Generieren, auf Grundlage eines ersten Maschinenlern- (ML) Modells, einer ersten Programmcodeanweisung, die mindestens der einen visuellen Aktion oder dem ersten visuellen Objekt entspricht; und  
ein Ausführen der ersten Programmcodeanweisung zum Modifizieren einer Visualisierungsschnittstelle der Tutor-Anwendung auf Grundlage der Texteingabe, die über die Chat-Schnittstelle empfangen wurde.

2. Verfahren nach Anspruch 1, wobei die erste Programmcodeanweisung eine Grafikanweisungsgrammatik aufweist, wobei das Verfahren ferner aufweist:  
ein Ausgeben der Visualisierungsschnittstelle zur Anzeige, wobei die Visualisierungsschnittstelle eine Mehrzahl von visuellen Objekten aufweist, einschließlich des ersten visuellen Objekts, wobei die Programmcodeanweisung mindestens eines von der Mehrzahl von visuellen Objekten modifiziert.

3. Verfahren nach Anspruch 2, ferner aufweisend:  
ein Empfangen einer Eingabe, die ein zweites visuelles Objekt von der Mehrzahl von Objekten modifiziert, das zur Anzeige über die Visualisierungsschnittstelle ausgegeben wird;  
ein Generieren einer zweiten Programmcodeanweisung zum Modifizieren des zweiten visuellen Objekts;  
ein Ausführen der zweiten Programmcodeanweisung zum Modifizieren des zweiten visuellen Objekts;  
ein Generieren einer Entität, auf Grundlage eines zweiten ML-Modells, die der zweiten Programmcodeanweisung entspricht;  
ein Generieren, auf Grundlage einer Zuordnung zwischen der Entität und einem Konzept, einer Anweisung in natürlicher Sprache; und  
ein Ausgeben der Anweisung in natürlicher Sprache über die Chat-Schnittstelle.

4. Verfahren nach Anspruch 3, ferner aufweisend, vor einem Empfangen der Texteingabe:  
ein Schulen des mindestens einen Klassifizierers, um die ersten und zweiten Maschinenlernmodelle auf Grundlage von Schulungsdaten zu generieren.

5. Verfahren nach Anspruch 4, wobei die Schulungsdaten aufweisen: (i) eine Mehrzahl von gekennzeichneten Programmcodeanweisungen, die zum

Generieren von Anweisungen in natürlicher Sprache verwendet werden, und (ii) eine Mehrzahl von gekennzeichneten Anweisungen in natürlicher Sprache, die zum Generieren von Programmcodeanweisungen verwendet werden.

6. Verfahren nach Anspruch 1, ferner aufweisend: ein Ausgeben einer ersten Nachricht über die Chat-Schnittstelle durch eine Chat-Engine; ein Klassifizieren eines Konzepts in der ersten Nachricht durch den mindestens einen Klassifizierer, der auf die erste Nachricht angewendet wird; ein Zuordnen des Konzepts in der ersten Nachricht zu einem Ereignis, das dem Konzept in der ersten Nachricht zugehörig ist; ein Generieren einer zweiten Programmcodeanweisung, die dem Ereignis entspricht, das dem Konzept in der ersten Nachricht zugehörig ist; und ein Ausführen der zweiten Programmcodeanweisung zum Modifizieren einer Visualisierungsschnittstelle der Tutor-Anwendung auf Grundlage der ersten Nachricht, die durch die Chat-Engine ausgegeben wurde.

7. Verfahren nach Anspruch 1, wobei die Texteingabe eine Antwort auf eine Frage aufweist, die über die Chat-Schnittstelle ausgegeben wurde.

8. Computerprogrammprodukt, aufweisend: ein durch einen Computer lesbares Speichermedium, auf dem ein durch einen Computer lesbarer Programmcode verkörpert ist, wobei der durch einen Computer lesbare Programmcode durch einen Prozessor ausführbar ist, um eine Operation auszuführen, die aufweist: ein Empfangen einer Texteingabe über eine Chat-Schnittstelle einer Tutor-Anwendung; ein Identifizieren eines Konzepts der Texteingabe durch mindestens einen Klassifizierer, der auf die Texteingabe angewendet wird; ein Zuordnen des Konzepts in der Texteingabe zu mindestens einer visuellen Aktion oder einem ersten visuellen Objekt; ein Generieren, auf Grundlage eines ersten Maschinenlern- (ML) Modells, einer ersten Programmcodeanweisung, die mindestens der einen visuellen Aktion oder dem ersten visuellen Objekt entspricht; und ein Ausführen der ersten Programmcodeanweisung zum Modifizieren einer Visualisierungsschnittstelle der Tutor-Anwendung auf Grundlage der Texteingabe, die über die Chat-Schnittstelle empfangen wurde.

9. Computerprogrammprodukt nach Anspruch 8, wobei die erste Programmcodeanweisung eine Grafikanweisungsgrammatik aufweist, wobei die Operation ferner aufweist: ein Ausgeben der Visualisierungsschnittstelle zur Anzeige, wobei die Visualisierungsschnittstelle eine Mehrzahl von visuellen Objekten aufweist, wobei die

Programmcodeanweisung mindestens eines von der Mehrzahl von visuellen Objekten modifiziert.

10. Computerprogrammprodukt nach Anspruch 9, ferner aufweisend:

ein Empfangen einer Eingabe, die mindestens eines von der Mehrzahl von Objekten modifiziert, die zur Anzeige über die Visualisierungsschnittstelle ausgegeben werden; ein Generieren einer zweiten Programmcodeanweisung zum Modifizieren des mindestens einen von der Mehrzahl von Objekten; ein Ausführen der zweiten Programmcodeanweisung zum Modifizieren des mindestens einen von der Mehrzahl von Objekten; ein Generieren einer Entität, auf Grundlage eines zweiten ML-Modells, die der zweiten Programmcodeanweisung entspricht; ein Generieren, auf Grundlage einer Zuordnung zwischen der Entität und einem Konzept, einer Anweisung in natürlicher Sprache; und ein Ausgeben der Anweisung in natürlicher Sprache über die Chat-Schnittstelle.

11. Computerprogrammprodukt nach Anspruch 10, wobei die Operation vor einem Empfangen der Texteingabe ferner aufweist:

ein Schulen des mindestens einen Klassifizierers, um das erste und zweite Maschinenlernmodell auf Grundlage von Schulungsdaten zu generieren.

12. Computerprogrammprodukt nach Anspruch 11, wobei die Schulungsdaten aufweisen: (i) eine Mehrzahl von gekennzeichneten Programmcodeanweisungen, die zum Generieren von Anweisungen in natürlicher Sprache verwendet werden, und (ii) eine Mehrzahl von gekennzeichneten Anweisungen in natürlicher Sprache, die zum Generieren von Programmcodeanweisungen verwendet werden.

13. Computerprogrammprodukt nach Anspruch 8, wobei die Operation ferner aufweist:

ein Ausgeben einer ersten Nachricht über die Chat-Schnittstelle durch eine Chat-Engine; ein Klassifizieren eines Konzepts in der ersten Nachricht durch den mindestens einen Klassifizierer, der auf die erste Nachricht angewendet wird; ein Zuordnen des Konzepts in der ersten Nachricht zu einem Ereignis, das dem Konzept in der ersten Nachricht zugehörig ist; ein Generieren einer zweiten Programmcodeanweisung, die dem Ereignis entspricht, das dem Konzept in der ersten Nachricht zugehörig ist; und ein Ausführen der zweiten Programmcodeanweisung zum Modifizieren einer Visualisierungsschnittstelle der Tutor-Anwendung auf Grundlage der ersten Nachricht, die durch die Chat-Engine ausgegeben wurde.

14. Computerprogrammprodukt nach Anspruch 8, wobei die Texteingabe eine Antwort auf eine Frage aufweist, die über die Chat-Schnittstelle ausgegeben wurde.

15. System, aufweisend:  
 einen Prozessor; und  
 einen Arbeitsspeicher, der eine oder mehrere Anweisungen speichert, die, wenn sie durch den Prozessor ausgeführt werden, eine Operation ausführen, die aufweist:  
 ein Empfangen einer Texteingabe über eine Chat-Schnittstelle einer Tutor-Anwendung;  
 ein Identifizieren eines Konzepts der Texteingabe durch mindestens einen Klassifizierer, der auf die Texteingabe angewendet wird;  
 ein Zuordnen des Konzepts in der Texteingabe zu mindestens einer visuellen Aktion oder einem ersten visuellen Objekt;  
 ein Generieren, auf Grundlage eines ersten Maschinenlern- (ML) Modells, einer ersten Programmcodeanweisung, die mindestens der einen visuellen Aktion oder dem ersten visuellen Objekt entspricht; und  
 ein Ausführen der ersten Programmcodeanweisung zum Modifizieren einer Visualisierungsschnittstelle der Tutor-Anwendung auf Grundlage der Texteingabe, die über die Chat-Schnittstelle empfangen wurde.

16. System nach Anspruch 15, wobei die erste Programmcodeanweisung eine Grafikanweisungsgrammatik aufweist, wobei die Operation ferner aufweist:  
 ein Ausgeben der Visualisierungsschnittstelle zur Anzeige, wobei die Visualisierungsschnittstelle eine Mehrzahl von visuellen Objekten aufweist, wobei die Programmcodeanweisung mindestens eines von der Mehrzahl von visuellen Objekten modifiziert.

17. System nach Anspruch 16, ferner aufweisend:  
 ein Empfangen einer Eingabe, die mindestens eines von der Mehrzahl von Objekten modifiziert, die zur Anzeige über die Visualisierungsschnittstelle ausgegeben werden;  
 ein Generieren einer zweiten Programmcodeanweisung zum Modifizieren des mindestens einen von der Mehrzahl von Objekten;  
 ein Ausführen der zweiten Programmcodeanweisung zum Modifizieren des mindestens einen von der Mehrzahl von Objekten;  
 ein Generieren einer Entität, auf Grundlage eines zweiten ML-Modells, die der zweiten Programmcodeanweisung entspricht;  
 ein Generieren, auf Grundlage einer Zuordnung zwischen der Entität und einem Konzept, einer Anweisung in natürlicher Sprache; und  
 ein Ausgeben der Anweisung in natürlicher Sprache über die Chat-Schnittstelle.

18. System nach Anspruch 17, wobei die Operation vor einem Empfangen der Texteingabe ferner aufweist:

ein Schulen des mindestens einen Klassifizierers, um das erste und zweite Maschinenlernmodell auf Grundlage von Schulungsdaten zu generieren.

19. System nach Anspruch 18, wobei die Schulungsdaten aufweisen: (i) eine Mehrzahl von gekennzeichneten Programmcodeanweisungen, die zum Generieren von Anweisungen in natürlicher Sprache verwendet werden, und (ii) eine Mehrzahl von gekennzeichneten Anweisungen in natürlicher Sprache, die zum Generieren von Programmcodeanweisungen verwendet werden.

20. System nach Anspruch 15, wobei die Texteingabe eine Antwort auf eine Frage aufweist, die über die Chat-Schnittstelle ausgegeben wurde, wobei die Operation ferner aufweist:  
 ein Ausgeben einer ersten Nachricht über die Chat-Schnittstelle durch eine Chat-Engine;  
 ein Klassifizieren eines Konzepts in der ersten Nachricht durch den mindestens einen Klassifizierer, der auf die erste Nachricht angewendet wird;  
 ein Zuordnen des Konzepts in der ersten Nachricht zu einem Ereignis, das dem Konzept in der ersten Nachricht zugehörig ist;  
 ein Generieren einer zweiten Programmcodeanweisung, die dem Ereignis entspricht, das dem Konzept in der ersten Nachricht zugehörig ist; und  
 ein Ausführen der zweiten Programmcodeanweisung zum Modifizieren einer Visualisierungsschnittstelle der Tutor-Anwendung auf Grundlage der ersten Nachricht, die durch die Chat-Engine ausgegeben wurde.

Es folgen 10 Seiten Zeichnungen

## Anhängende Zeichnungen

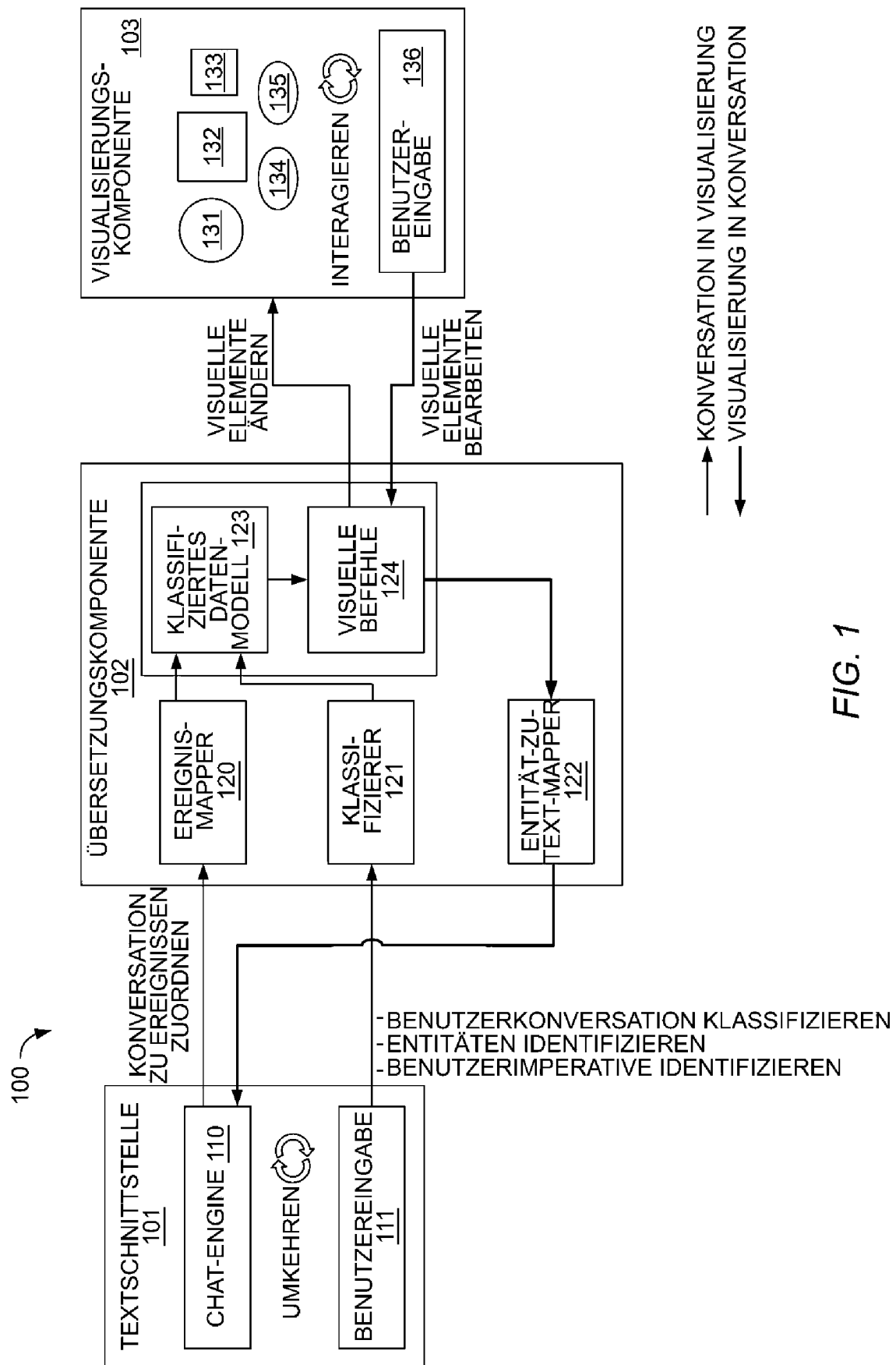
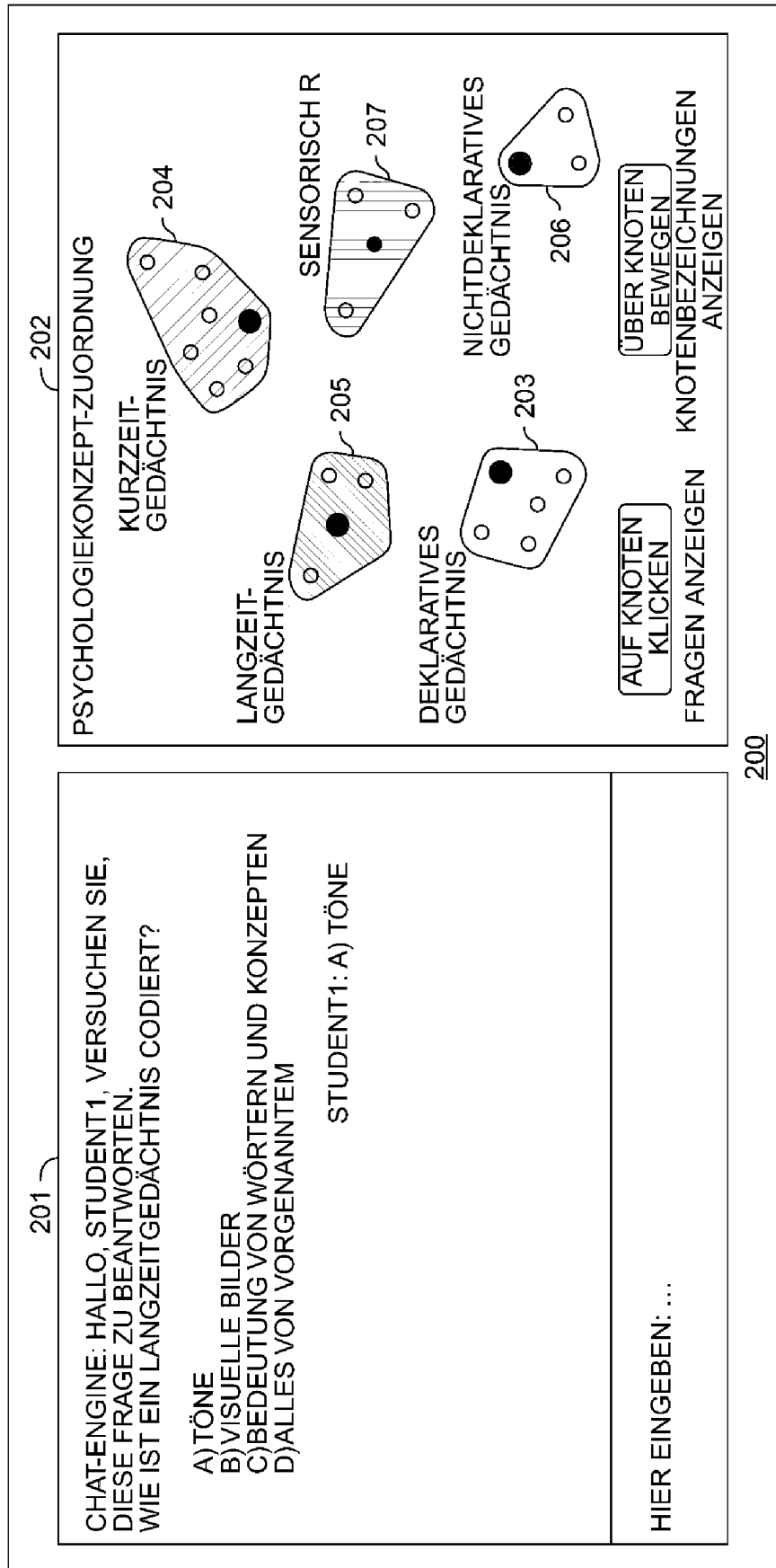


FIG. 1



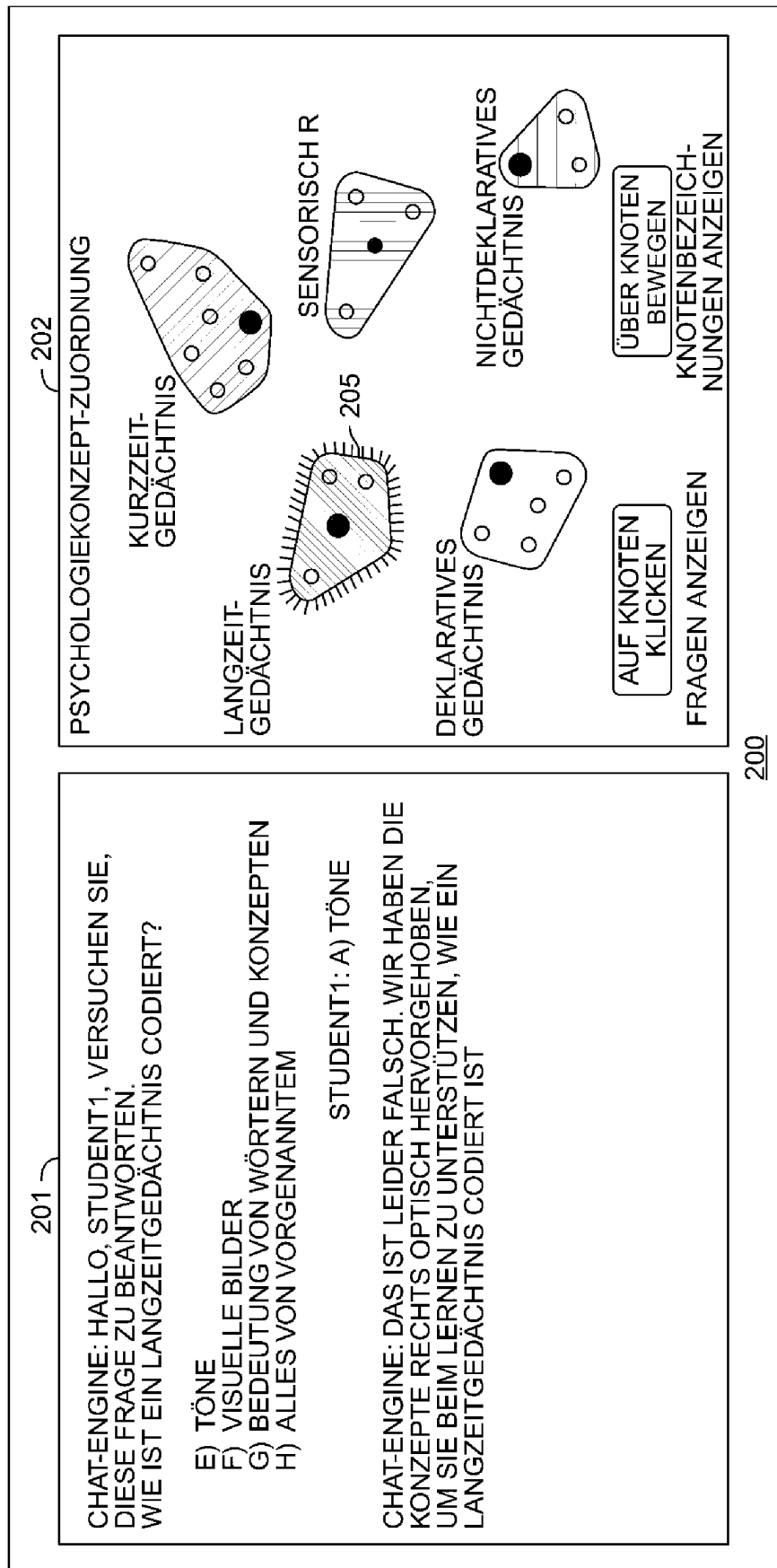


FIG. 2B

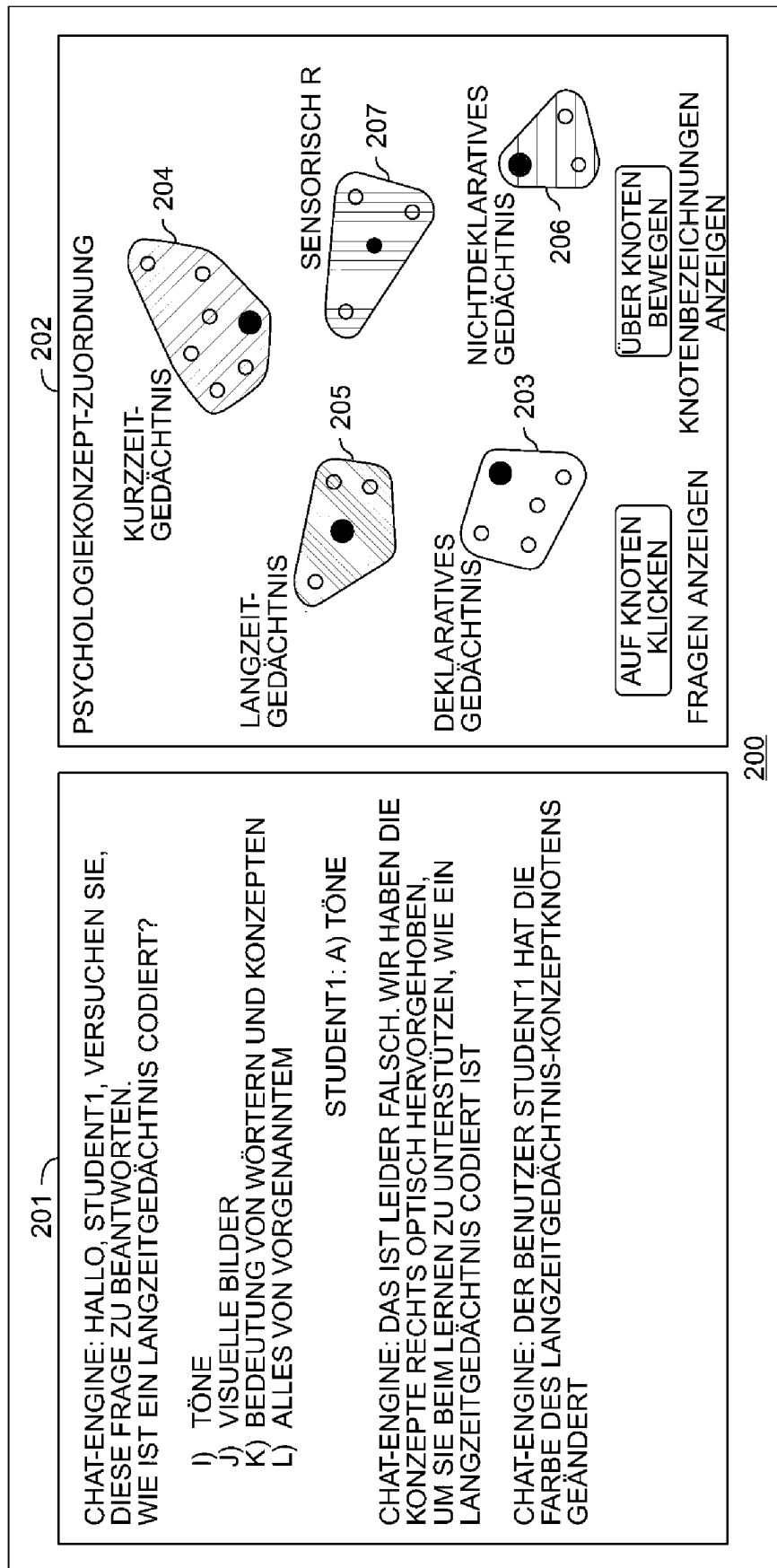


FIG. 2C



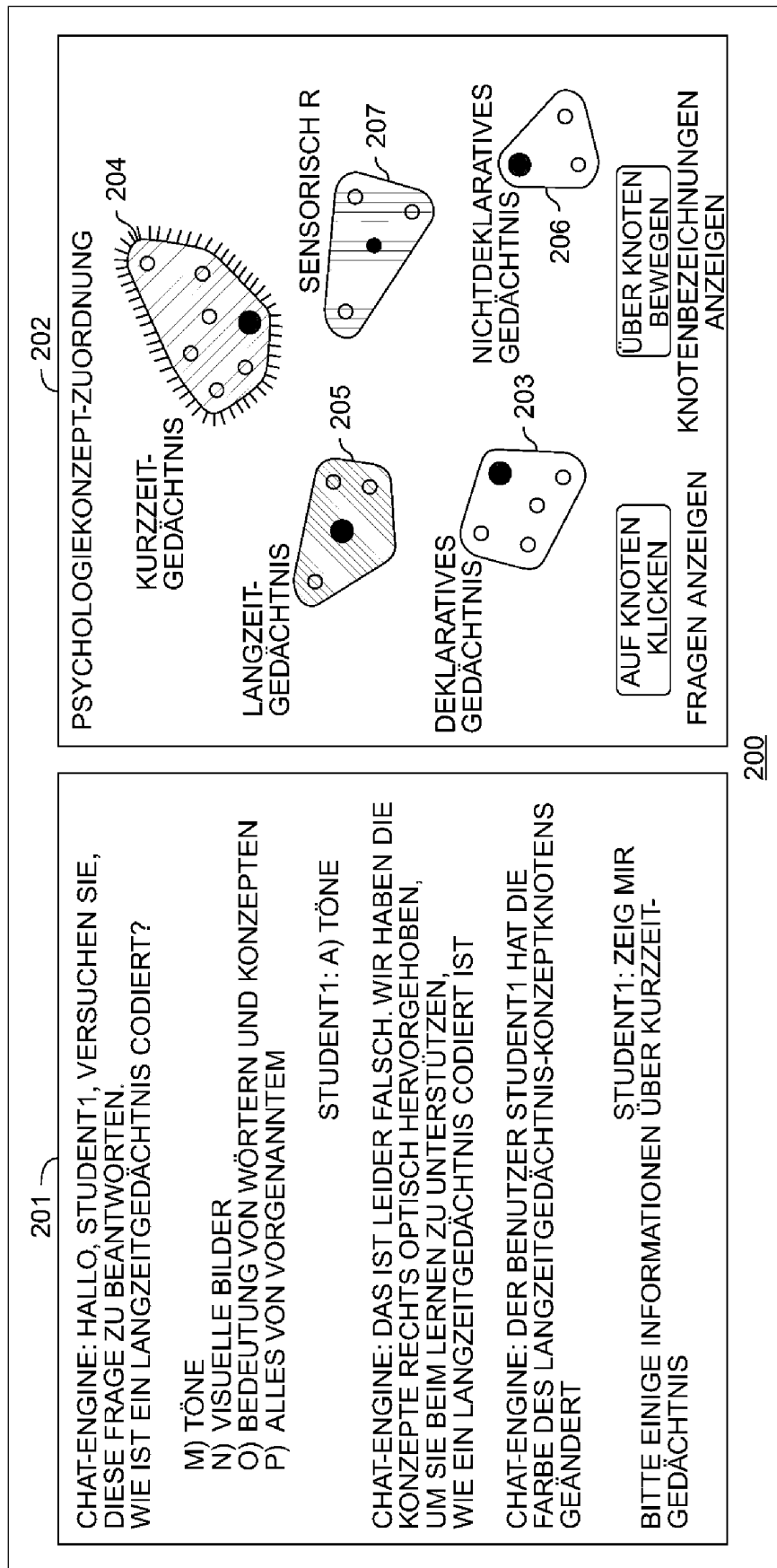


FIG. 2D

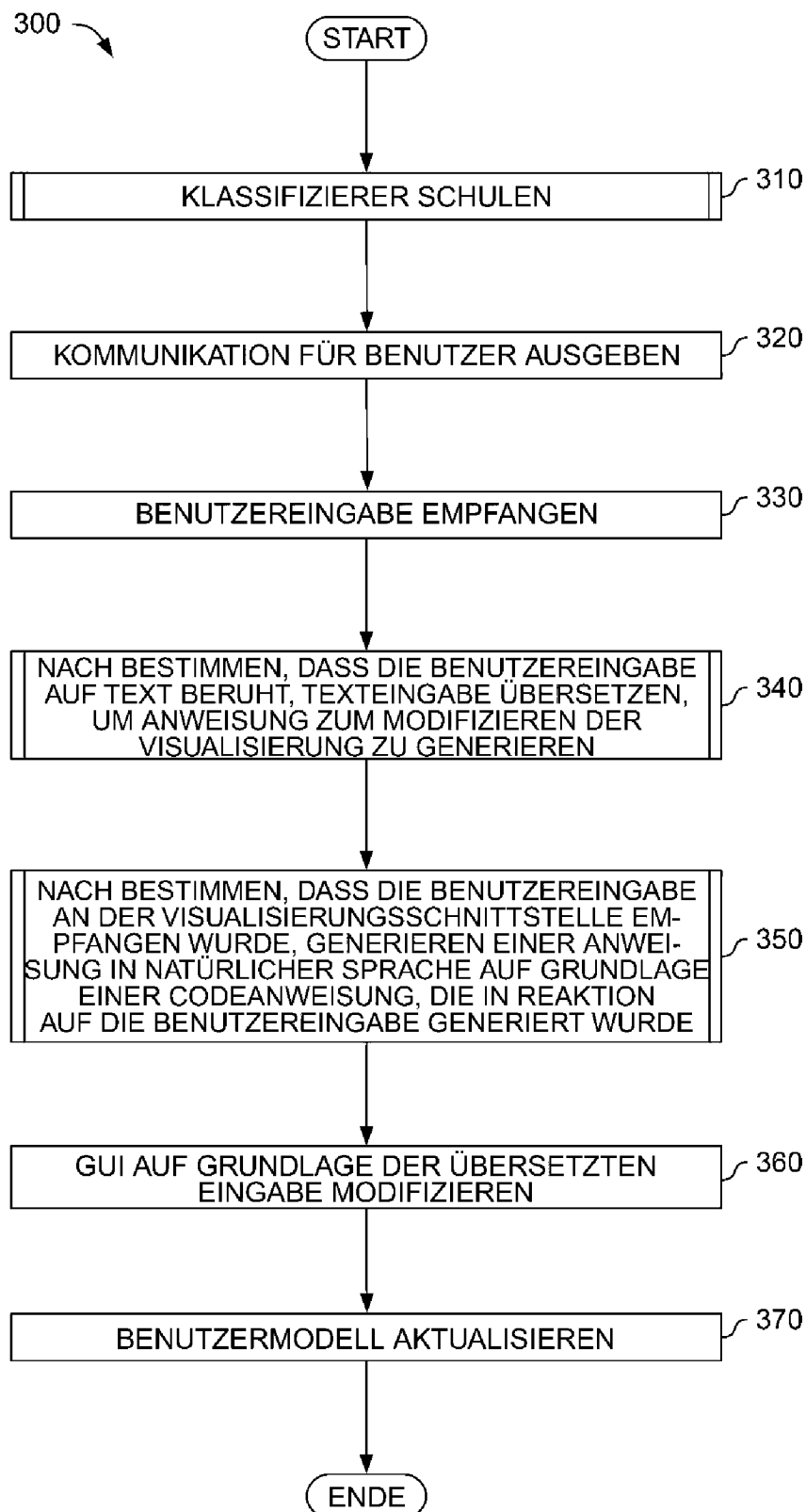
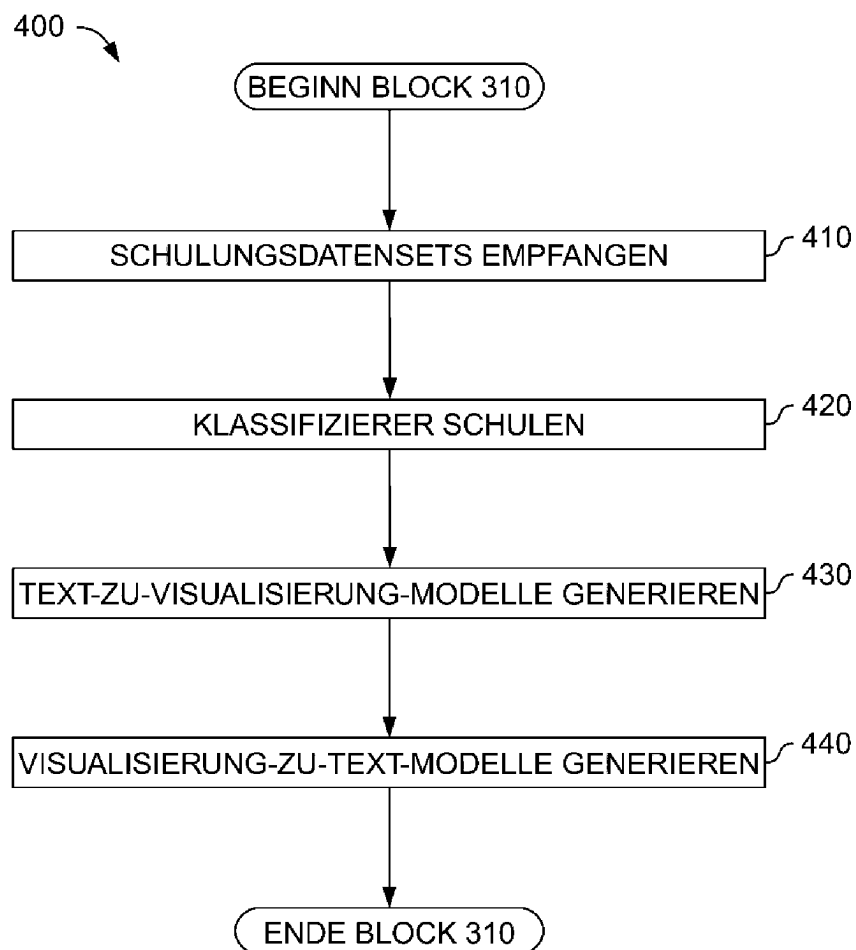


FIG. 3



**FIG. 4**

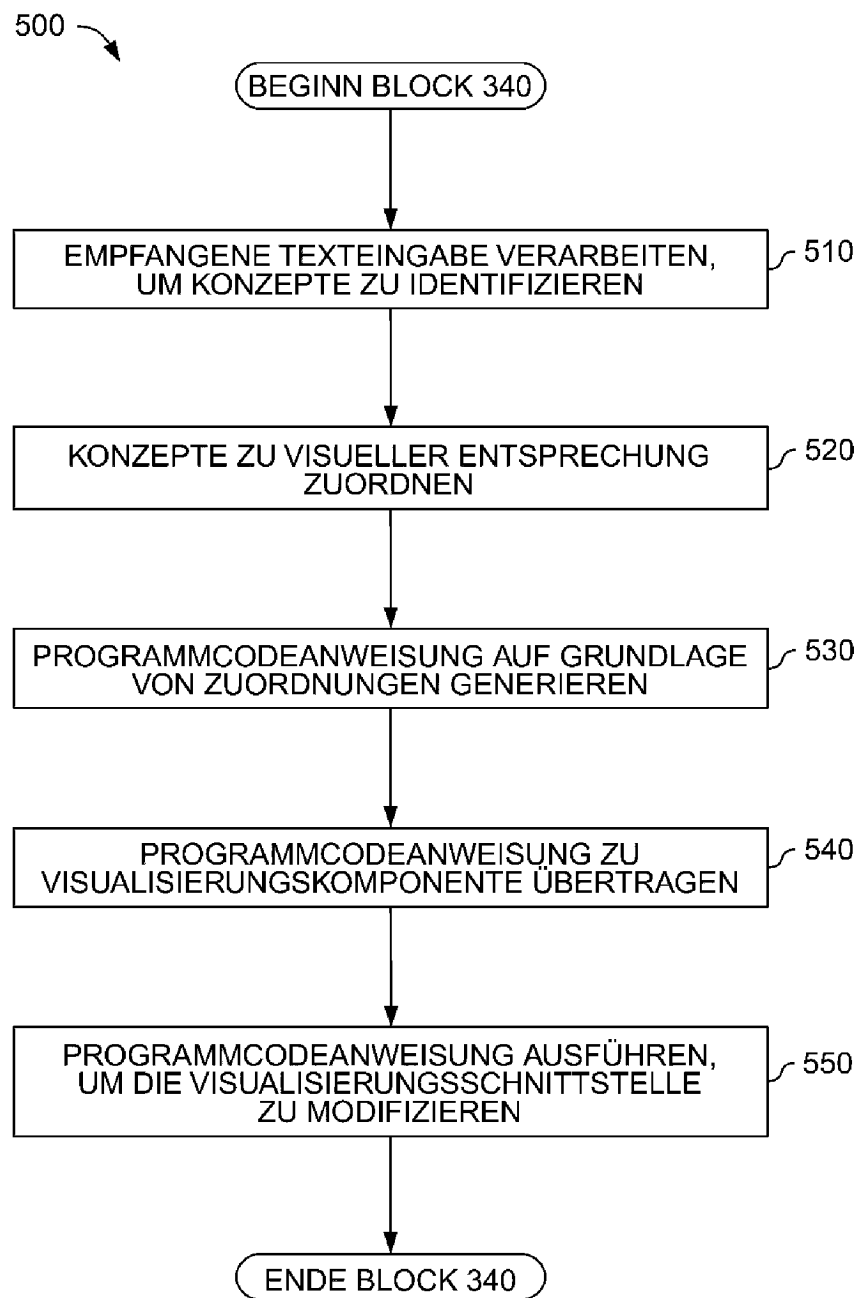


FIG. 5

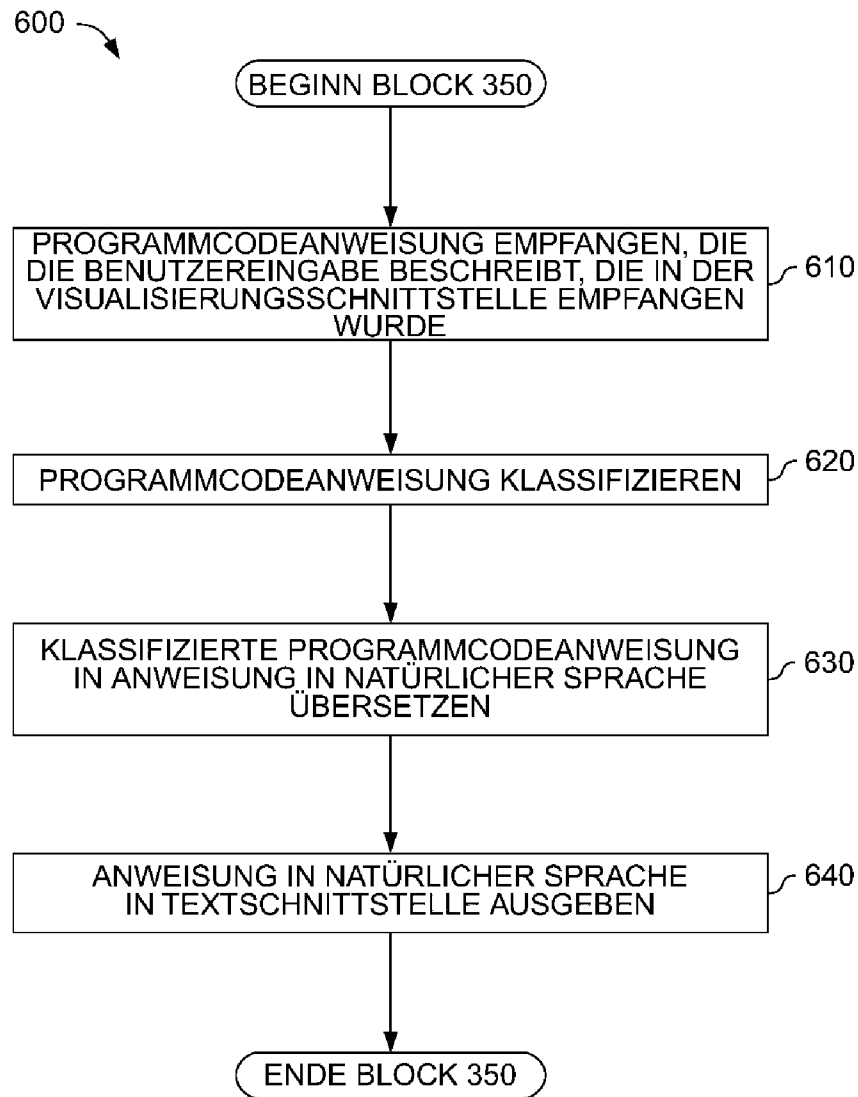


FIG. 6

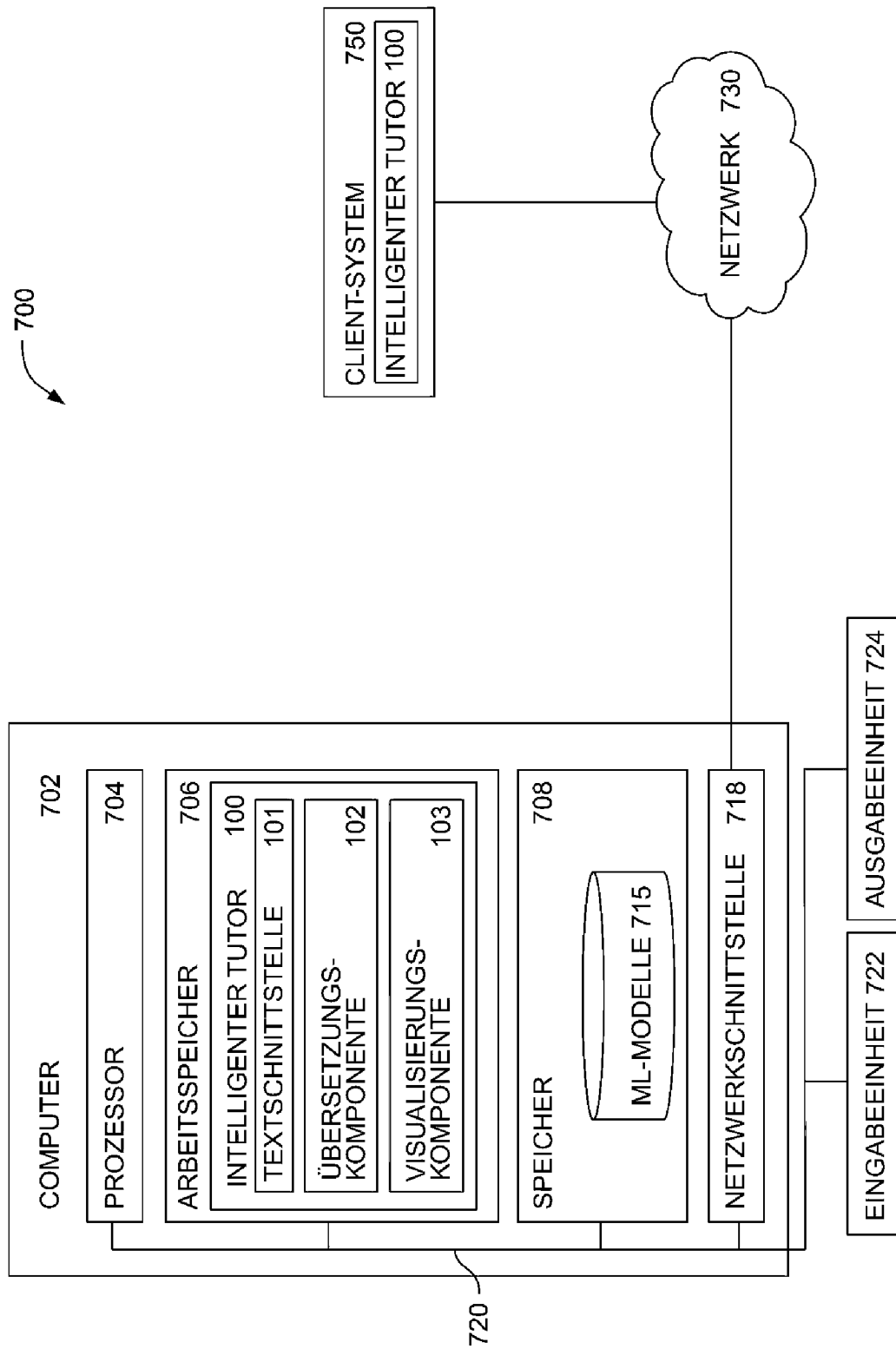


FIG. 7