Analyzing the Potential Impact of Robotic Process Automation and Artificial Intelligence on the Education Sector: Opportunities, Challenges, and Implications

Chase Hunter

CIS 4950

5/4/23

10:00 - 10:30 am

Introduction

- For generations, education has relied on traditional teaching methods as the standard. The lecture-style classroom, textbooks, and written exams have been entrenched in the educational landscape. While these methods have demonstrated their efficacy, they possess certain limitations
- Since the COVID-19 pandemic, students have become more accustomed to online learning, and now it is often seen as a regular part of student education
- With this in mind, it opens the door to many new opportunities in the education sector, including implementing robotic process automation and artificial intelligence without overwhelming the teachers and students

What is Robotic Process Automation (RPA)?

- Robotic Process Automation, otherwise known as RPA, involves using computer software to automate repetitive tasks, with the overall goal to increase efficiency and reduce errors

What is it used for?

- With RPA solutions on the rise, it is actually seen in almost all industries, but more so those that include frequent repetitive tasks
    - RPA is most commonly seen in insurance, banking, finance, healthcare and telecommunications

How is RPA different from AI?

- While the two may seem similar, they perform in different ways and have different uses
- RPA uses structured inputs and logic that is provided by the user
- AI uses unstructured inputs and typically will develop its own logic
- Although they are not the same, RPA and AI can perform quite efficiently when paired together

What is the potential benefit of RPA and AI implementation in the education sector?

The use of RPA in education has been shown to have several potential benefits, including increased efficiency and accuracy in educational processes, enhanced interaction and engagement between students and teachers, and improved educational outcomes.

Notable Points

- RPA can improve efficiency and effectiveness in the education sector, particularly administrative processes
- Using RPA can help teachers save time and increase productivity, allowing them to focus more on teaching and student learning
- RPA can provide personalized learning experiences and support teachers in providing tailored feedback to students
- The integration of RPA in education has been shown to increase teacher satisfaction and motivation
- Studies have shown the potential benefits of RPA in reducing errors and increasing accuracy in tasks such as grading and record keeping

What are some specific areas RPA and AI implementation could improve in education?

- Any area with frequent repetitive tasks could benefit from RPA and AI implementation

- In the following slides, I will be covering four specific areas:
  - Scheduling
  - Student Learning Experience
  - Financial Aid Processing
  - Student Information Management

Scheduling

Potential Benefits:

- Enhanced schedule optimization
- Real-time schedule adjustments
- Predictive analytics
- Reduced administrative workload
- Optimize schedules more effectively than traditional algorithms
- More customized and efficient schedules
- Enhanced student and teacher satisfaction
- More efficient educational experience

Potential Risks:

- Perpetuating Discrimination
- Reducing role of human educators and administrators
- Technical issues that could cause disruptions

Student Learning Experience

Potential Benefits:

- Personalized Learning
- Intelligent Tutoring
- Customized Content Curation
- Automated Grading
- Predictive Analytics

Potential Risks:

- Hindered critical thinking and problem-solving skills
- Privacy and security concerns
- Biases perpetuated in the learning experience
- Unknown future for students learning via AI/Automation practices

Financial Aid Processing

Potential Benefits:

- Increased efficiency, accuracy, personalization, and fraud detection
- Automated verification and eligibility determination
- Reduced workload for financial aid officers
- Accelerated Financial Aid process
- Personalized recommendations depending on specific financial situations

Potential Risks:

- Biased decision-making due to inaccurate and unfair data collection and analysis
- System algorithms may exclude students with certain unique circumstances
- Complex technology may leave students confused when navigating application process
- Reduce opportunities for human employees and their professional development
- Privacy and security concerns regarding personal information

Student Information Management

Potential Benefits:

- Automated administrative tasks
- Predictive analytics
- Early warning systems for at-risk students
- Identify patterns and trends to provide customized learning experiences
- Use information collected to anticipate future outcomes

Potential Risks:

- Accuracy of data gathered
- Confidentiality and safety of personal student information
- Unintended prejudice or unfairness
- Job cuts for human employees who manage pre-existing student information system

Recap

- Many potential benefits to RPA and AI implementation in these four education systems
- Also many potential risks to consider
- Does the benefit outweigh the risks?
- Does the risks outweigh the benefits?

Risk-Benefit Analysis

Scheduling:

- AI algorithms and process automation can optimize school schedules and reduce workload
- But also poses risks such as perpetuating discriminatory biases and reducing human involvement
- Mitigation measures, such as transparency, accountability, and human oversight are necessary for fair and effective scheduling process
- Benefits of implementation outweigh the potential risks

Student Learning Experience:

- RPA and AI implementations can offer benefits like personalized learning and automated grading
- But implementation also poses risks like hindering critical thinking
- Transparency and collaboration with human educators are crucial to mitigate risks
- Implementing RPA and AI in the student learning experience is worthwhile as long as the risks are adequately addressed, and benefits are applied responsibly

Financial Aid Processing:

- RPA and AI implementations in financial aid processing may have biased decision-making, decreased access, and job loss issues
- Careful consideration required to determine if benefits outweigh the risks, including mitigating biases, and protecting privacy and security.
- Responsible use is crucial, balancing benefits and risks, implementing mitigation measures, and fostering critical thinking alongside AI-based learning.

Student Information Management:

- RPA and AI in student information management can offer personalized learning, simplified tasks, and early warning systems.
- Risks include but may not be limited to, data accuracy, privacy concerns, and potential bias
- Implementing RPA and AI in education may be worthwhile if risks are addressed and technology is used responsibly
- Benefits outweigh risks if AI is designed to be inclusive, transparent, and protective

Implementation

- Not many examples of RPA implementation in education for a variety of reasons
  - Organizations are unsure what tasks to focus their efforts on
- Although, some RPA implementation has been attempted in the education sector with vastly positive results
  - See Institute of Engineering and Technology Wiley examples on next slide
- Successful implementation shows substantial increase in operational efficiency
- While implementation results are positive, development timelines are long and costly
  - Additional time needed for stabilization and training where necessary

Prior RPA Implementation Examples & Results

TABLE 8 Post-RPA process automation and operational efficiency analysis for the HR assistant bot

Old process	New process	Development time	Operational efficiency	Relevance, impact and limitations
Manual scanning of job portals, collating applicable job listings and sharing with concerned stakeholders	Automatic retrieval of job listings from multiple sources, checking applicability, classifying and categorising opportunities, and sharing with concerned stakeholders	1 Developer, 80 h effort for development and testing	Estimated cumulative time saving of 200 h of manual effort per year at the institution level	Relevance: HIGH; impact: HIGH; limitations: complex logic for determining eligibility of students for each job listing and validity of listings

TABLE 11 Post-RPA process automation and operational efficiency analysis for the teaching assistant bot

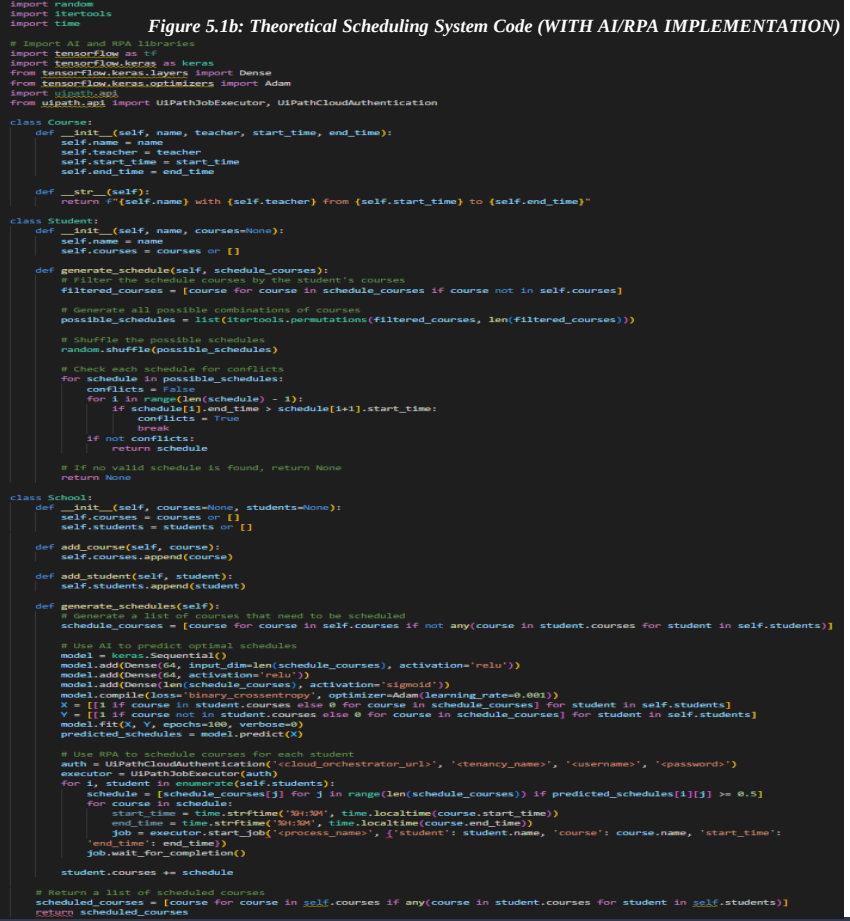
Old process	New process	Development time	Operational efficiency	Relevance, impact and limitations
Manual in nature	Automatic retrieval, parsing, list creation, plagiarism check, student communication, and updating student records in ERP	2 Developers, 100 h effort for development and testing	Estimated cumulative time saving of 6–8 h of manual effort per month per faculty	Relevance: HIGH; impact: HIGH; limitations: sometimes detection of figures in assignments is a challenge for the RPA bot

- Two successful implementation examples from the researches at IET Wiley
- See significant gains in operational efficiency
- While efficiency gains are high, so are development times

Implementation Cont.

- In slide 7-10, I discussed the potential benefits and risks of RPA and AI implementation in four common school systems
- Using this theoretical information, I created a prompt for OpenAI's GPT-3.5 model to attempt to write code for these four systems
- The following slides is the result I received when explaining a school scheduling system, financial aid processing system, and a student information management system, all with RPA and AI implementation
    - *Note:* The student learning experience was left out of this exercise due to the need for personalization depending on the specific situation

Figure 5.1b: Theoretical Scheduling System Code (WITH AI/RPA IMPLEMENTATION)



```python
import random
import itertools
import time

# Import AI and RPA libraries
import tensorflow as tf
import tensorflow.keras as keras
from tensorflow.keras.layers import Dense
from tensorflow.keras.optimizers import Adam
import pyautogui
from uipath.api import UiPathJobExecutor, UiPathCloudAuthentication

class Course:
    def __init__(self, name, teacher, start_time, end_time):
        self.name = name
        self.teacher = teacher
        self.start_time = start_time
        self.end_time = end_time

    def __str__(self):
        return f"{self.name} with {self.teacher} from {self.start_time} to {self.end_time}"

class Student:
    def __init__(self, name, courses=None):
        self.name = name
        self.courses = courses or []

    def generate_schedule(self, schedule_courses):
        # Filter the schedule courses by the student's courses
        filtered_courses = [course for course in schedule_courses if course not in self.courses]

        # Generate all possible combinations of courses
        possible_schedules = list(itertools.permutations(filtered_courses, len(filtered_courses)))

        # Shuffle the possible schedules
        random.shuffle(possible_schedules)

        # Check each schedule for conflicts
        for schedule in possible_schedules:
            conflicts = False
            for i in range(len(schedule) - 1):
                if schedule[i].end_time > schedule[i+1].start_time:
                    conflicts = True
                    break
            if not conflicts:
                return schedule

        # If no valid schedule is found, return None
        return None

class School:
    def __init__(self, courses=None, students=None):
        self.courses = courses or []
        self.students = students or []

    def add_course(self, course):
        self.courses.append(course)

    def add_student(self, student):
        self.students.append(student)

    def generate_schedules(self):
        # Generate a list of courses that need to be scheduled
        schedule_courses = [course for course in self.courses if not any(course in student.courses for student in self.students)]

        # Use AI to predict optimal schedules
        model = keras.Sequential()
        model.add(Dense(64, input_dim=len(schedule_courses), activation='relu'))
        model.add(Dense(64, activation='relu'))
        model.add(Dense(len(schedule_courses), activation='sigmoid'))
        model.compile(loss='binary_crossentropy', optimizer=Adam(learning_rate=0.001))
        X = [[1 if course in student.courses else 0 for course in schedule_courses] for student in self.students]
        Y = [[1 if course not in student.courses else 0 for course in schedule_courses] for student in self.students]
        model.fit(X, Y, epochs=100, verbose=0)
        predicted_schedules = model.predict(X)

        # Use RPA to schedule courses for each student
        auth = UiPathCloudAuthentication('<cloud_orchestrator_url>', '<tenancy_name>', '<username>', '<password>')
        executor = UiPathJobExecutor(auth)
        for i, student in enumerate(self.students):
            schedule = [schedule_courses[j] for j in range(len(schedule_courses)) if predicted_schedules[i][j] >= 0.5]
            for course in schedule:
                start_time = time.strftime('%H:%M', time.localtime(course.start_time))
                end_time = time.strftime('%H:%M', time.localtime(course.end_time))
                job = executor.start_job('<process_name>', {'student': student.name, 'course': course.name, 'start_time':
            'end_time': end_time})
            job.wait_for_completion()

        student.courses += schedule

    # Return a list of scheduled courses
    scheduled_courses = [course for course in self.courses if any(course in student.courses for student in self.students)]
    return scheduled_courses
```

- The system includes three classes: Course, Student, and School
- The Course class represents a course with its name, teacher, start time, and end time
- The Student class represents a student with their name and courses they are enrolled in
- The School class represents a school with a list of courses and students
- The generate_schedule method in the Student class generates a schedule for a student based on the courses that are not already taken by the student
    - It generates all possible combinations of courses and checks for conflicts in the schedule. If a valid schedule is found, it is returned. If not, None is returned
- The generate_schedules method in the School class generates schedules for all students using AI to predict optimal schedules for each student
- It creates a neural network model that takes in the courses each student is enrolled in as input and predicts the courses that they should take to maximize their schedule
- RPA is used to schedule courses for each student, It creates a UiPathJobExecutor instance and starts a job for each course to be scheduled
- The method returns a list of scheduled courses that are taken by at least one student

Theoretical Scheduling System (With AI/RPA Implementation)

5.2b - Theoretical Financial Aid Processing System Code (WITH AI/RPA IMPLEMENTATION)

```
import pandas as pd
from sklearn.linear_model import LinearRegression
import rpa as r

class Student:
    def __init__(self, name, income, gpa):
        self.name = name
        self.income = income
        self.gpa = gpa
        self.financial_aid = None

class FinancialAidProcessor:
    def __init__(self, students):
        self.students = students

    def process_aid(self):
        # Collect and preprocess data
        df = pd.DataFrame([(s.income, s.gpa) for s in self.students], columns=['Income', 'GPA'])
        df['Financial Aid'] = [s.financial_aid for s in self.students]

        # Train an AI model to predict financial aid
        model = LinearRegression()
        model.fit(df[['Income', 'GPA']], df['Financial Aid'])

        # Use the model to predict financial aid for each student
        for student in self.students:
            predicted_aid = model.predict([[student.income, student.gpa]])[0]

            # Update student record with predicted financial aid information
            student.financial_aid = predicted_aid

        # Use RPA to update the student records in the school's database
        r.init()
        r.login(username='school_admin', password='12345')
        for student in self.students:
            r.update_student_record(student.name, student.financial_aid)
        r.logout()

        # Return a list of students with updated financial aid information
        return self.students

# Example usage
john = Student("John", 25000, 3.7)
jane = Student("Jane", 35000, 3.2)
joe = Student("Joe", 15000, 3.9)
sarah = Student("Sarah", 45000, 2.8)

students = [john, jane, joe, sarah]
processor = FinancialAidProcessor(students)
processed_students = processor.process_aid()
for student in processed_students:
    print(f"{student.name} has been awarded ${student.financial_aid} in financial aid.")
```

- This code is for a school financial aid processing system that uses AI and RPA to predict and award financial aid to students
- The Student class represents each student and their information, including income, GPA, and financial aid amount
- The FinancialAidProcessor class processes financial aid for students using income and GPA information provided
- Pandas library is used to collect and preprocess data for each student
- Linear Regression AI model is used to predict the financial aid amount for each student based on income and GPA
- RPA is used to update student records in the school's database with predicted financial aid information
- The code returns a list of all students with their updated financial aid information
- Example usage of the code creates four student objects with their respective information and passes them to the FinancialAidProcessor class.

Theoretical Financial Aid Processing System (With AI/RPA Implementation)

5.3b - Theoretical Student Information Management System Code (WITH AI/RPA IMPLEMENTATION

```
import pandas as pd
import numpy as np
import tensorflow as tf
import rpa as r

# Load student data
student_data = pd.read_csv('student_data.csv')

# Define early warning system
def early_warning_system(student_data):
    # Identify at-risk students
    at_risk_students = student_data[student_data['GPA'] < 2.0]
    for index, row in at_risk_students.iterrows():
        # Send alert to counselor or advisor
        r.email(to=row['Advisor'], subject='At-Risk Student Alert', body='Dear ' + row['Advisor'] + ',\n\nThis is to inform you that ' + row['StudentName'] + ' is at risk of falling behind or dropping out due to low GPA. Please reach out to them and provide appropriate support and guidance.\n\nSincerely,\nStudent Information Management System')
    return at_risk_students

# Define enrollment function
def enroll_student(student_data, student_name, course_code):
    # Check if student is eligible for course
    if student_data[student_data['StudentName'] == student_name]['GPA'].values[0] >= 2.0:
        # Add course to student's transcript
        student_data.loc[student_data['StudentName'] == student_name, course_code] = 'Enrolled'
        # Update student's GPA
        student_data.loc[student_data['StudentName'] == student_name, 'GPA'] = np.mean(student_data.loc[student_data['StudentName'] == student_name, ['Math', 'Science', 'English', 'History']].values)
        # Send confirmation email to student
        r.email(to=student_data[student_data['StudentName'] == student_name]['Email'].values[0], subject='Enrollment Confirmation', body='Dear ' + student_name + ',\n\nThis is to confirm that you have been enrolled in ' + course_code + '. Please refer to your updated transcript for details.\n\nSincerely,\nStudent Information Management System')
    else:
        # Send email to counselor or advisor for intervention
        r.email(to=student_data[student_data['StudentName'] == student_name]['Advisor'].values[0], subject='Intervention Required', body='Dear ' + student_data[student_data['StudentName'] == student_name]['Advisor'].values[0] + ',\n\nThis is to inform you that ' + student_name + ' has attempted to enroll in ' + course_code + ', but their GPA falls below the minimum requirement. Please provide
        appropriate support and guidance.\n\nSincerely,\nStudent Information Management System')
    return student_data

# Define transcript retrieval function
def get_transcript(student_data, student_name):
    # Retrieve student's transcript
    transcript = student_data.loc[student_data['StudentName'] == student_name, ['Math', 'Science', 'English', 'History']].transpose().reset_index()
    # Rename columns
    transcript.columns = ['Course', 'Grade']
    # Convert grades to letter grades
    transcript['Grade'] = np.where(transcript['Grade'] >= 90, 'A', np.where(transcript['Grade'] >= 80, 'B', np.where(transcript['Grade'] >= 70, 'C', np.where(transcript['Grade'] >= 60, 'D', 'F'))))
    return transcript

# Define AI model
model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(128, input_shape=(4,), activation='relu'),
    tf.keras.layers.Dense(64, activation='relu')

# Compile the model
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
    return model

# Define a function to train the AI model
def train_model(X_train, y_train, epochs=10):
    # Instantiate the model
    model = create_model()
    # Fit the model to the training data
    history = model.fit(X_train, y_train, epochs=epochs)
    return model, history

# Define a function to predict at-risk students
def predict_at_risk_students(student_data, model):
    # Prepare the data for prediction
    X = student_data[['GPA', 'Math', 'Science', 'English', 'History']].values
    # Make predictions
    y_pred = model.predict(X)
    # Add the predictions to the student data
    student_data['Prediction'] = y_pred.round().astype(int)
    # Identify at-risk students
    at_risk_students = student_data[student_data['Prediction'] == 1]
    for index, row in at_risk_students.iterrows():
        # Send alert to counselor or advisor
        r.email(to=row['Advisor'], subject='At-Risk Student Alert', body='Dear ' + row['Advisor'] + ',\n\nThis is to inform you that ' + row['StudentName'] + ' is at risk of falling behind or dropping out based on our AI predictions. Please reach out to them and provide appropriate support and guidance.\n\nSincerely,\nStudent Information Management System')
    return at_risk_students

# Train the AI model
X_train = student_data[['GPA', 'Math', 'Science', 'English', 'History']].values
y_train = student_data['AtRisk'].values.reshape(-1, 1)
model, history = train_model(X_train, y_train, epochs=10)

# Predict at-risk students
at_risk_students = predict_at_risk_students(student_data, model)
```

Code explanation on next slide..

- This theoretical student information management system manages student information including names, GPA, courses, and grades, and has several useful features such as:
    - Early warning system for at-risk students
    - Enrollment function
    - Transcript retrieval function
    - AI model

- The early warning system identifies at-risk students with a GPA below 2.0 and sends alerts to their counselor or advisor using the RPA library

- The enrollment function checks eligibility for a course based on GPA, adds it to the student's transcript, and updates their GPA. Confirmation emails are sent to students and counselor intervention emails are sent if the student is not eligible

- The transcript retrieval function retrieves a student's transcript, renames columns, and converts grades to letter grades

- The AI model is a sequential model with two dense layers, it predicts at-risk students based on GPA and subject grades, the model is trained using the training data, and its predictions are added to student data

- The predict_at_risk_students function sends alerts to counselors or advisors for each student predicted to be at risk of falling behind or dropping out based on AI predictions

- This system is designed to help students stay on track with their education and provide appropriate support to those at risk of falling behind or dropping out

Theoretical Student Information Management System (With AI/RPA Implementation) Cont...

Implementation Recap

- While the previously shown code does not currently produce any quantifiable output, it does create a general idea of the work necessary to create the proposed system
- Since there is no use case where these systems will be implemented, I will be using successful implementation timelines from researchers at The Institute of Engineering and Technology (IET) Wiley as a general reference
- Eleven different implementation exercises were completed by researchers at IET Wiley (2 shown in slide 14), with development timelines ranging from 30 to 100 hours depending on the task
    - Using this information, I will estimate an average development and testing timeline of 74 hours for my proposed systems
- Using the same eleven tests, I can estimate a timeline for testing and stabilization of my proposed systems of around 5 weeks
- To estimate increases in operational efficiency, I will once again find an average from the known increases in operational efficiency found by researchers at IET Wiley
    - Using this information, I will estimate an average increase of around 134 hours per month for my proposed systems
- Because these systems are all so different, I can not 100% rely on these estimates, but it does create a general idea of development timelines, testing timelines, and potential increases in operational efficiency
- On the next slide I will cover the final important piece to be considered before implementation, ethical considerations

Ethical Considerations

Privacy:

- Privacy is a potential ethical issue with proposed education technologies
- Proposed tech has the ability to not only gather but also analyze large amounts of sensitive information
- Information could potentially be used in an inappropriate manner or create new vulnerabilities in unforeseen cyber attacks
- Careful consideration is needed from education executives and policymakers
- Proper guidelines are necessary for data privacy and security

Bias:

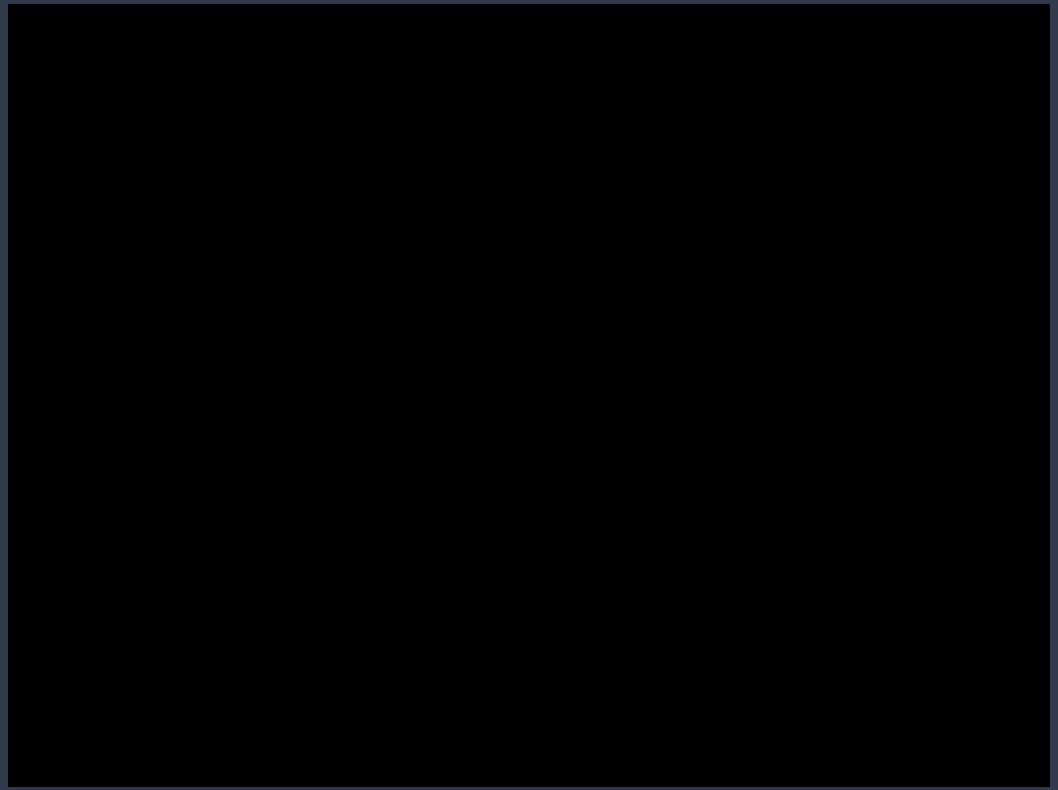
- Technologies can only be as unbiased as the data they are originally trained on
- Data could reflect societal biases which would lead to AI technologies replicating said bias
- Could lead to issues in areas like college admissions or student assessment
- Retraining of RPA and AI algorithms necessary to minimize bias

Job Displacement:

- While true that RPA and AI technologies can massively improve efficiency in administrative tasks
- RPA and AI technologies also have the ability to replace jobs from humans that worked said position prior to implementation
- These are vital implications to consider when implementing these technologies into the education workforce
- Strategies must be put in place to mitigate the negative side effects of job displacement

Technical Demo

Basic Schedule UI with Automation



Conclusion

- RPA and AI have the potential to revolutionize the education sector with significant advancements
- Integrating RPA and AI in education can improve efficiency, personalization, and learning outcomes
- Ethical and responsible utilization of these technologies is crucial, and concerns surrounding data privacy and security must be addressed
- Further research and exploration is necessary to fully comprehend the potential of RPA and AI in education and ensure their responsible and effective deployment
- Benefits of RPA and AI must be balanced with their ethical considerations to promote their successful implementation in education

Thank You

Chase M. Hunter
Department of Computer Science and Information Systems
University of North Georgia
Dahlonega, GA, United States of America
cmhunt2679@ung.edu