Семинар "Педагогика ИИ"

Дата: 20.08.2025 10:00

Место проведения: НИУ «МЭИ», г. Москва, ул. Красноказарменная, д. 13с3, аудитория М-101.

Участники: Роман Сергеевич Куликов (НИУ «МЭИ»), Олег Вадимович Глухов (НИУ «МЭИ»), Павел Юрьевич Анучин (НИУ «МЭИ»), Владимир Владимирович Чистяков (НИУ «МЭИ»), Константин Евгеньевич Бошков (НИУ «МЭИ»), Челышев Эдуард Артурович(НИУ «МЭИ»), Константин Александрович Петров (НИЦ "Курчатовский институт" - НИИСИ), Александр Павлович Еремеев(НИУ «МЭИ»), Зурова Мария Александровна (НИУ «МЭИ»), Шамиль Алиевич Оцоков(НИУ «МЭИ»), Степан (179 школа)

Повестка:

1) Тема основной дискуссии «Проектирование ИИ-ассистентов инженеров»:

- Доклад рабочей группы по ИИ при ИРЭ (МЭИ) «Обзор методов обучения больших языковых моделей под специализированные задачи»;

2) Выбор тематики следующего семинара.

Олег Вадимович Глухов:

Добрый день, коллеги! У нас проходит уже пятый семинар. Здесь сегодня есть новые представители. Я специально для вас расскажу, что у нас семинар носит открытый характер. Могут быть представители различных заинтересованных организаций. Он регулярный формат имеет. Каждые две недели собираемся и пишем аудиозапись, по которой потом пишется стенограмма. Сейчас мы успели уже по двум семинарам это сделать и в ближайшие дни выложим на портал МЭИ. Эту информацию можно будет передавать своим коллегам для того, чтобы как можно больше участников мы здесь видели. И всех призываю по возможности в следующий раз приходить очно. Это будет и нам всем сподручнее, и дискуссия будет гораздо живее. Ну и, собственно, давайте по традиции представимся. Меня зовут Олег Вадимович Глухов. Я являюсь организатором этого семинара. Представляю Московский Энергетический Институт. Уже с 2018 года занимаюсь использованием методов искусственного интеллекта в

своих изысканиях. И вот давайте, коллеги, по часовой стрелке представимся. Расскажите кратко, как вы соприкасаетесь с ИИ в своей работе. Сначала давайте за столом, а потом коллег по онлайну.

Павел Юрьевич Анучин:

Коллеги, добрый день! Меня зовут Павел Юрьевич Анучин, я сотрудник кафедры радиотехнических систем Московского энергетического института. Здесь мы занимаемся разработкой прикладных применений и для отрасли, и не только, где как раз таки апробируем подходы отчасти те, которые в рамках данного семинара мы вырабатываем, и какие-то интересные находки мы тоже здесь как бы рассматриваем и применяем потом в работе. Поэтому я в прямом смысле заинтересован, чтобы мы с вами здесь обсуждали не только какие-то моменты и больше как пространно, а еще и моменты прикладные, технические. Поэтому, в принципе, являюсь непосредственным участником отрасли. Спасибо.

Владимир Владимирович Чистяков:

Здравствуйте, коллеги. Меня зовут Владимир Владимирович Чистяков. Я являюсь сотрудником кафедры РТС НИУ МЭИ. Я являюсь нахожусь в составе рабочей группы по разработке искусственного интеллекта. И, в принципе, у меня интересы такие же, как и у Павла, потому что мы, можно сказать, в одной команде двигаемся.

Константин Евгеньевич Бошков:

Коллеги, здравствуйте. Меня зовут Константин Евгеньевич. Я являюсь сотрудником кафедры РТС НИУ МЭИ и нахожусь в той же рабочей группе, что и мои коллеги. Спасибо.

Челышев Эдуард Артурович:

Здравствуйте, коллеги. Челышев Эдуард Артурович, ассистент кафедры вычислительных машин системы сетей. Собственно, интересно было бы послушать последние, скажем так, новости в сфере искусственного интеллекта и выработать стратегию развития, в том числе профессионального, и существования в эту непростую интеллектуальную эпоху.

Роман Сергеевич Куликов:

Роман Сергеевич Куликов, директор института радиотехники и электроники НИУ МЭИ. Я сейчас решаю задачу применения искусственного интеллекта как ассистента-инженера.

Мы пробуем обучить большую языковую модель программам нашим образовательным по радиотехнике и электронике, чтобы получить такой интеллектуальный САПР для повышения производительности разработчика радиотехники. Для того, чтобы мы коллегами могли выработать наиболее эффективные подходы для решения этой задачи.

Константин Александрович Петров:

Здравствуйте, меня зовут Константин Александрович Петров, я из НИИСИ РАН. Я здесь как человек, который сталкивается по своей работе с использованием машинного обучения искусственного интеллекта в области проектирования микроэлектроники. И еще как преподаватель кафедры электроники Московского физического института, который, так сказать, готовит студентов для нашей отрасли микроэлектроники и тоже интересуется, чему их сейчас надо учить. Спасибо.

Александр Павлович Еремеев:

Я профессор кафедры прикладной математики и искусственного интеллекта. Руковожу научной группой имени Дмитрия Александровича Поспелова по тематике конструирования интеллектуальных систем поддержки принятия решений реального времени и руководитель гранта РНФ по этому же направлению. Ну, с 2003 по 2020 год я заведовал кафедрой прикладной математики. Сейчас она... в одной тематике - искусственный интеллект. Хочу отметить, что наши кафедры и мы в целом были одни из лидеров по подготовке специалистов в области искусственного интеллекта, то есть где-то с 90-х годов, конца 90-х годов у нас работал Дмитрий Александрович Поспелов. Нам удалось провести профиль подготовки специалистов в области искусственного интеллекта. Сейчас мы готовим этих специалистов в рамках магистратуры по программе искусственный интеллект. Я читаю ряд базовых курсов по этой дисциплины - основа искусственного интеллекта это четвертый курс балкалавриата и магистром интеллектуальные системы поддержки принятия решений. Экспертная система, то есть опыт подготовки специалистов в области искусственного интеллекта. Тематика хорошая, и то, что вот мы участвовали, ну и многие, наверное, из вас, 25-го, на семинаре лидирующих вузов в области подготовки специалистов, где был ряд докладов из МЭИ. То есть, Волошин докладывал, докладывал Вишняков и так далее. Я считаю этот семинар очень полезным, и то, что МЭИ находится, ну, я контактирую с коллегами МГТУ, МИФИ, Фистеха, среди лидеров по подготовке специалистов в области искусственного интеллекта, это очень хорошо. И вот этот семинар и направлен на то, чтобы мы, так сказать, эту планку не снижали.

Зурова Мария Александровна:

Здравствуйте еще раз. Меня зовут Зурова Мария Александровна. Я ассистент кафедры ВМСС. Искусственный интеллект – это в области моих личных научных интересов. К сожалению, ни в какой научной группе я не состою, поэтому этот семинар для меня в том числе возможность поиска единомышленников.

Шамиль Алиевич Оцоков:

Оцоков Шамиль Алиевич, я доцент кафедры вычислительных машин системы сетей. Я входил в состав участников которые реализовывали проект по оценки деятельности кафедры нашего университета по систему ЕФКУЭМ с использованием искусственного интеллекта. Спасибо.

Степан:

Добрый день, я стажер АО SignalTech. Мы используем искусственный интеллект для разработки программного обеспечения, а также искусственный интеллект в области моих научных интересов. Я просто как студент любительский изучаю эту технологию.

Олег Вадимович Глухов:

Тогда если все представились, давайте перейдем к теме нашей дискуссии. Значит, у нас сегодня основная тема – это проектирование и ассистентов-инженеров. Значит, видится так, что это будет цикл семинаров, посвященный этой тематике. И сегодня мы рабочей группой по искусственному интеллекту при ИРЭ МЭИ подготовили доклад, который звучит как “Обзор методов обучения больших языковых моделей под специализированные задачи”. У нас там есть презентация большая, я думаю, мы сегодня весь семинар будем ее обсуждать. И предлагается так, что будет докладывать у нас Владимир, а можно задавать вопросы прямо по ходу доклада.

Владимир Владимирович Чистяков:

Ну да, потому что презентация получилась довольно большая, и пока мы ее закончим, уже вопросы забудутся, и настроение упадет, может у кого-то, может у кого-то поднимется, не знаю.

Давайте тогда разделим как-то. Я в голове своей разделю презентацию на кластеры и буду говорить стоп, и тогда кто хочет, может высказываться.

Еще раз повторюсь, меня зовут Владимир Владимирович Чистяков, и наша сегодняшняя тема доклада – это обзор методов обучения бюджетных языковых моделей под специализированной задаче. Скажу сразу, что не все методы мы рассматривали, потому что если рассматривать все методы, то мы здесь до декабря будем где-то сидеть. Я условно, конечно, говорю, но мы именно рассматривали

методы, делали акцент больше на тех методах, которые мы сами использовали в своих разработках. В общем, для начала нужно понять, что такое большая языковая модель.

Здесь дано определение большой языковой модели, но на самом деле это определение можно и к малым языковым моделям применить. Что большое, что малое, вопрос достаточно, как я сказал, философский, потому что в 2018 году, я условно могу сказать, 20 миллиардов параметров считалась большая модель, сейчас уже 700, даже больше есть. Поэтому из того, что я видел, никакой стандартизации именно нет. Есть условное разделение, сверхбольшие модели считаются от 200 миллиардов примерно там и выше. Большие языковые модели – это от 200 миллиардов параметров до 50, я имею в виду вниз если идти. Малые языковые модели – это от 50 до 1 миллиарда. И ниже – это сверхмалые. Но я скажу так, что на самом деле это все условное обозначение. Если вы скажете, допустим, что 100 миллиардов – это сверхбольшая языковая модель, то я думаю, что никто вам ничего плохого не скажет.

Когда мы начинали разработку, или какой-то человек, допустим, хочет применить в своей сфере искусственный интеллект. И вот перед ним становится страгический выбор. А что мне делать? Либо разработать искусственный интеллект с нуля, либо адаптировать уже какое-либо решение, которое сделали наши коллеги из Америки, или из Европы, или также Яндекс, допустим из России.

Первое, если мы решили разрабатывать с нуля. Что мы должны понимать, если мы будем разрабатывать модель с нуля? На что нам придется делать акценты? Первое, это затраты, то есть это стоит больших денег. Понятное дело, что если брать государство, то для государства это небольшие деньги, но если мы берем именно какие-то там сферы производства и какой-нибудь там, не знаю, ну человек, допустим, там хочет разработать в своем гараже что-либо, то для него это очень большие затраты получаются. Нужен гигантский объем данных, чтобы с нуля обучить модель, это все прекрасно знают, вычислительные мощности, которые тоже упираются в затраты, ресурсы и так далее, ну и эксперты, глубокие эксперты в теме, которые могут помочь.

То есть, грубо говоря, для большинства компаний, разработчиков обучение с нуля является экономически и технически нецелесообразным.

Я сейчас приведу именно в цифрах, как это все показано.

Есть табличка, которую мы взяли у наших коллег. Правда, здесь немного старые модели, несмотря на то, что статья относительно, если брать научную сферу, не прям старая, но искусственный интеллект развивается так быстро, что сегодня выпускайте статью, через неделю она уже относительно старая. Поэтому здесь, если вы видите, ну, слева модели, название модель, их параметры там, на каких датасетах, размерах датасетов они обучались. И вот интересно, есть hardware specification, это именно вычислительные ресурсы, которые тратились на то, чтобы обучить модель с нуля. И я примерные и привел цифры, то есть где-то 67 миллионов долларов, где-то написано часы. Часы модели обучали, получается, брали в аренду мощности и обучали. И

написано количество дней, сколько это все заняло. В принципе, вот эти суммы, которые написаны справа, это только на вычислительный блок. получается, только на вычисления. То есть здесь не учитывалось затраты на зарплаты, на сборку датасета и так далее. То есть здесь именно цифры, сколько стоит развернуть такое окружение для обучения модели.

Олег Вадимович Глухов:

Можно я тут дополню? Если коллеги видите, здесь Представлена модель GPT-3, то есть это уже, по сути, два прошлых поколения назад. Сейчас уже есть GPT-5 и аналогичные модели, и, соответственно, скорее всего, там затраченные мощности больше. Хотя все мы знаем про опыт дипсика, который, поменяв подход, сэкономил на несколько порядков.

Владимир Владимирович Чистяков:

У кого-то есть какие-нибудь вопросы?

Роман Сергеевич Куликов:

А сколько потратил DeepSeek в сравнении с этим, известно?

Владимир Владимирович Чистяков:

DeepSeek, как я читал, потратил, по-моему, полтора или два миллиона долларов.

Но при этом сейчас они вроде как разворачивают на своих чипах. И пока неизвестно, сколько они на следующую версию потратят.

Роман Сергеевич Куликов:

Там другого рода проблема. Техническая переработанная. А вот этих параметров токенов, объема параметров и длительности обучения известны?

Владимир Владимирович Чистяков:

У кого? У Deepseek? Нет. По крайней мере, я этим вопросом не задавался пока что.

Роман Сергеевич Куликов:

Очень плохо, потому что вот эта агитка, вы все в нее верите. И она у вас убивает в волю к победе.

Владимир Владимирович Чистяков:

Нет, не убивает.

Роман Сергеевич Куликов:

Убивает, ты же сам говоришь. Очень дорого и нецелесообразно. Но мы не смотрели как и не будем смотреть. Мы будем верить в эту агитку, что это нам недоступно.

Владимир Владимирович Чистяков:

Нет, мы будем смотреть. Это очень интересно.

Константин Александрович Петров:

Ну да, согласен. Хорошее замечание. Я хотел бы добавить, Роман Сергеевич, что была информация, что дипсик на самом деле не так, чтобы сильно дешевле, там не на порядке дешевле. Если считать как-то более правильно, есть мнение в индустрии, что не так у них дешево получилось, как они пишут в промо-материалах.

Роман Сергеевич Куликов:

Хотелось бы взглянуть не на вот эти охи и ахи из инстаграма, а все-таки на нормальную спецификацию техническую, вот в этой таблице, например, сопоставить. Вот это было бы гораздо интереснее, чем слушать агитки про то, что это нам никогда не доступно.

Владимир Владимирович Чистяков:

Замечания хорошие, мы обязательно это решим. Проработаем эти вещи.

Олег Вадимович Глухов:

Мы для этого и собираемся, чтобы получить обратную связь!

Владимир Владимирович Чистяков:

В общем, какое есть решение, если нет доступа к таким финансам. Можно взять уже готовую модель, открытую, и ее дообучить. Грубо говоря, зачем изобретать колесо и использовать велосипед? Просто возьми уже колесо готовое и делай велосипед. Поэтому именно такой подход мы использовали. То есть мы взяли готовую уже модель и ее дообучили.

Дальше мы покажем. Это таблица открытых моделей.

Мы привели модели, не все их использовали, мы использовали другую модель, но чтобы показать, какие модели на какие задачи больше ориентированы. Ну, и то, что метод QLora, который мы позже тоже рассмотрим, Можно многие из этих моделей обучить. Не надо суперкомпьютер собирать.

Можно на домашний ПК обучить модель своим задачам.

Методы адаптации.

Есть три подхода. Это мы сейчас говорим именно про, когда мы берем готовую модель. Дипсик или Квен. Что мы можем сделать? Мы можем полностью, полное дообучение, то есть полностью, грубо говоря, переобучить модель. И вот здесь как-то говорится, что это как взять выпускника вуза, он, допустим, был ориентирован на программирование, мы ему начинаем гуманитарные задачи давать, то есть это очень долго, да, это в том последствии эффективно, но это и затратно. Есть вот второй метод, то есть это мы берем, допустим, ну я тоже так скажу условно, человек знает, там, Python, C++, а мы ему говорим, давай учить JavaScript. При этом примерно понимает структуру, понимает программирование, и ему будет проще выучить новый язык условно программирования, чем если дать человеку, который занимался строительством, ему сказать, давай программируй. Третий метод – это RAG. Мы модель ничему не обучаем. То есть мы ее обучаем общаться с библиотекой. Мы ей даем справочник какой-нибудь или еще что-нибудь такое именно в формате RAG, и она к нему обращается и выдает нам данные.

Есть какие-нибудь вопросы?

Константин Александрович Петров:

Вопрос такой, а как это все технически делается, и по затратам, что каждый из этого требует? Особенно в сравнении второго и третьего, потому что первый, я так понимаю, это то, что вообще нам недоступно.

Владимир Владимирович Чистяков:

Про этот первый метод, да, мы дальше прям подробно расскажем, какой нужен объем вычислительных ресурсов. Про полное обучение, скажу вам честно, мы как бы читали, да, узнавали. Те ресурсы, которые у нас сейчас есть, их как бы недостаточно для этого. Я не могу вам больше сейчас сказать именно про первый метод, потому что мы не углублялись именно руками туда.

Олег Вадимович Глухов:

Первый метод, скажу так, он на самом деле был представлен в той таблице, которую мы обсуждали. То есть это буквально переобучение с нуля. То есть мы архитектуру берем устоявшуюся, известную, и фактически на своем датасете полностью ее переобучаем. То есть там все коэффициенты будут изменены.

Александр Павлович Еремеев:

Вот, ну, наверное, вы, коллеги, слушали выступление недавнее, известного специалиста в области Игоря Ашманова, то есть член, по-моему, общественной палаты по информационной безопасности, и он предупреждал, что вообще-то использование западных систем, это означает вся информация будет там у держатели и разработчиков этих систем. Ну, понятно, если это в процессе как обучения студентов, может быть, это и не очень критично, но если это используется в серьезных работах, в

том числе и двойного назначения, то это вообще-то довольно опасный подход, ориентация на западные продукты. Тем более, по большинству из них мы исходников не имеем. Вот этот вопрос вы как-то решаете? То есть вы все-таки берете их пакеты или это... свои разработки. Но, насколько я знаю, таких серьезных систем своих на большие языковые модели пока, в общем-то, нет. Таких универсальных достаточно.

Роман Сергеевич Куликов:

Давайте я отвечу. Мы, когда ставили вопрос об этом, как раз и уделили первое очередное внимание тому, где лежит код нейросети, где он работает. Если брать популярную в свое время LLM-чат GPT, то она не дает вам исходный код, она дает вам интерфейс. А если брать такую модель, как DeepSeek, у нее есть возможность локализации на вашем собственном сервере. В дальнейшем работаете в своем собственном контуре. Поэтому этот вопрос, естественно, вставал в первую очередь и на него есть ответы, что некоторые из моделей, они имеют возможность локализации на вашем частном сервере. Олег, я правильно говорю?

Олег Вадимович Глухов:

Все правильно, да. То есть здесь произошло, может быть, наложение понятий. Мы сейчас не какие-то сервисы используем, мы буквально скачиваем архитектуру предобученную. То есть там уже настроены коэффициенты.

То есть это как в распознавании образов. Вот тоже есть примеры моделей, там YOLO, например, для детекции объектов. Она обучена на всех картинках, которые у нее были до этого. Есть такая технология переноса обучения, когда замораживаются первые слои и обучают только конечные, которые необходимы для выполнения вертоуровневых заданий. Но здесь немножко по-другому будет, про это мы расскажем. Но смысл в том, что все эти модели, они локализуются.

Роман Сергеевич Куликов:

Не все эти, а которые здесь представлены на слайде.

Олег Вадимович Глухов:

Да, да, и в том числе, которые мы будем рассматривать, да, они локализуются на нашем железе, обучаются в том числе на нашем железе, да, обучаются. Вот, то есть тут полностью с точки зрения информационной безопасности такой правильный подход.

Роман Сергеевич Куликов:

То есть мы в дальнейшем работаем с теми моделями, которые локализуемы.

Шамиль Алиевич Оцоков:

Можно такой еще вопрос? Вы сказали про DeepSeek, она представлена, она локализуется, но у них есть урезанная версия, которая подходит для слабого железа. А нормальная версия, которая может достаточно хорошо и на русском языке отвечать на вопросы, она требует очень мощных ресурсов. Как вы используете API? Вопрос такой, что вы используете API для работы или вы выбрали слабую версию и пробовали ее заземлить?

Владимир Владимирович Чистяков:

Мы сначала выбрали слабую версию, потому что мы примерно понимали ресурсы, то есть использовался мой ноутбук, я примерно понимал, что он может потянуть, что может не потянуть, и мы решили сначала, ну, не сразу большой, ну, я даже могу сейчас сказать, то есть у меня мой ноутбук, у него там стоит RTX 4050, laptop-овская версия, то есть это сразу, как говорим, урезанная, 6 гигабайт видеопамяти, то есть он не прям сильно позволяет что-либо включать. Он потянет модель максимум наверное это где-то 7 миллиардов параметров вот так плюс-минус и мы решили так как 7 миллиардов параметров он слишком долго думает я взял 1.3 миллиарда дипсик параметров и в принципе на русском языке он разговаривает отлично даже может даже немножко лучше, чем я. Понятное дело, что если проводить много тестов, задавать ему какие-то вопросы, которые он не знает, не понимает, это маленькая модель относительно, то он, может быть, не справится. Но также и большие нейронки, даже тот же Chat-GPT 5, несмотря на то, что она максимально крутая, как нам представляют и так далее, она тоже, ну, в первый день мы там тестили, проверяли ее, она тоже может допускать ошибки, так что... Надеюсь, ответили.

Шамиль Алиевич Оцоков:

Да, ответили. Просто я почему-то спросил. Я пробовал сработать с урезанной версией DeepSeek. Честно говоря, очень слабые результаты, какие-то слабые ответы, бессмысленные. Часто галлюцинации происходят.

Владимир Владимирович Чистяков:

Там еще зависит от того, насколько маленькая версия, то есть, допустим, недавно вышла Gemma, Gemma 270 миллионов. Раньше была гонка за тем, там, больше параметров, больше знаний, сейчас большие компании стараются развивать именно маленькие модели, чтобы сделать это как... мобильные решения.

Олег Вадимович Глухов:

Есть пример, в робототехнике сейчас развивается такое направление, как VLM, это Visual Language Model. Идея какая, что на робота ставится небольшая языковая модель, которая позволяет ему по той сенсорике, которую он принимает, имитировать некое осмысление того, что перед ним находится. То есть считается, что LLM – это некий

сейчас строительный блок универсальный, который позволит понимать, почему, допустим, вот эта пицца, которую наблюдают на столе, как она здесь оказалась, чья она, откуда пришла и так далее.

Владимир Владимирович Чистяков:

Для чего нужно полное обучение. Это когда, допустим, модель у вас, ну, здесь написано, что она обучалась на старых каких-то данных, там, 2010 года, тут вы моргнули уже 2030 год. То есть данные изменились, нужно их обновить. Поэтому лучше использовать полное обучение. Второй момент, это если вы ее обучали на какой-то информации, потом к вам начальник пришел, сказал, эта информация является конфиденциальной, или она была конфиденциальной, то лучше всего это полностью переобучить модель, чтобы она вообще про это все забыла. Вот так. Да. Параметры эффективного обучения, это, грубо говоря, у нас есть мозг, мы ему сверху приделываем кластеры, как нейролинк, там, условно, его мозг.

То есть мы не трогаем весь мозг, мы ему сверху еще добавляем знания отдельно, условно.

Сейчас я просто хочу это все обобщить, чтобы прям все это не рассказывать. Ну, допустим, тоже про код. Я вам там приводил пример. То есть модель умеет писать код, но на языках, на питон,си плюс-плюс, а вы хотите, чтобы она на верилоге писала. То есть удобнее сделать это и не полностью переобучать, то, что она синтаксис кода примерно понимает, проще это ее дообучить, получается. У вас есть модель, которая умеет переводить текст с русского на английский, а вы хотите, чтобы она переводила на бенгальский, к примеру, еще. Вы же не будете полностью переобучать модель, вы просто ей добавите еще возможность, так скажем, понимать и распознавать бенгальский язык.

Олег Вадимович Глухов:

Это важно, да, только при условии, если самые первые слои при формировании вот этих осмысленных конструкций, они одинаковы, что в одном и другом языке.

Владимир Владимирович Чистяков:

Ну да, то есть здесь очень важен именно такой момент, что если, допустим, вы берете модель, которая специализируется на, например, знание английского языка, с английского на русский переводить, а вы говорите, давай будешь мне финансы считать. Этого делать не нужно, нужно брать именно ту модель которая ну похожа на тот проект который вы хотите сделать если вы хотите обучать модель коду значит вы должны взять именно модель предобученную коду, хотите модель чтобы она вам распознавала изображение, лучше взять модель который уже предобучена на это.

Далее, RAG. Это как раз то, что я говорил. То есть у нас есть моделька. Она не запоминает. Она просто открывает книжку по запросу и выдает данные, которые вы попросили.

Очень, наверное, полезно, я так думаю, если вы пишите методички, в принципе, небольшие, то RAG-формат, наверное, удобен, потому что вы можете вносить туда изменения. И модель сразу перестроится. Она сама не перестроится, она будет выдавать тот запрос, который вам нужен. Здесь это понятно.

Олег Вадимович Глухов:

Это касается той информации, которая не должна интерпретироваться каким-то субъектом. Вот, она должна быть объективной, то есть она заложена в какую-то базу уже, мы к ней просто подключаемся. Это как ответить на вопрос, какой у меня курс доллара. Модель, если у нее не будет доступа к бирже, просто попробует на основании своего представления мира спрогнозировать, но это будет не тот же самый курс доллара, который сегодня на бирже.

Владимир Владимирович Чистяков:

Здесь я привел, так скажем, минусы, но, наверное, интереснее нам больше следующая таблица. Разница между двумя подходами

Роман Сергеевич Куликов:

Тут приведено медленно, быстрее . О каком времени идет речь?

Олег Вадимович Глухов:

Давайте я отвечу. Значит, если мы говорим про одно и то же железо, то здесь все упирается в то, насколько большая база данных у RAG. Если это, допустим, 10 тысяч примеров каких-то, которых мы скормили до этого, сформировали эту базу, ну, какой-то учебник, допустим, мы туда запихнули, то это, допустим, плюс пару секунд ответ. Чем больше она будет расширяться, тем больше это будет расти. Не знаю, линейно или не линейно, но, в общем, тут все будет зависеть просто от размера этой базы.

Александр Павлович:

У меня такой вопрос относительно железа. Может быть, я чего-то не расслышал. Насколько я знаю, GPT-5, по-моему, 4,5, где-то этот уровень. У них обучение… Железо – это 86 тысяч ядер компьютер, то есть это мощный суперкомпьютер. И то обучение длится довольно долго, если мы хотим достаточно универсальную систему получить. Там терабайтные подключаются базы. Вот все-таки вы на какое железо ориентируетесь, чтобы действительно вот то, что говорил профессор Оцоков, да,

чтобы достаточно мощная система была, так сказать, а не игрушечный вариант такой демонстрационный. Ведь нужны очень мощные суммера, которых, насколько я знаю, ну, может, у меня не вся информация, так сказать, мало кто обладает. Даже наши суперкомпьютеры типа Чебышев и так далее, это то, что сделано. Они, ну, хорошо, если сейчас где-то в сотку входят, так сказать.

Владимир Владимирович Чистяков:

Но вы сейчас говорите про полное дообучение. Туда нужны такие ресурсы. Если мы берем уже готовую модель и сверху ей условно приделываем адаптер, мы дальше это рассмотрим, то тогда ресурсы намного меньше требуются. На обычном компьютере вы можете обучить. И срок зависит от вашего, условно, датасета и ваших настроек, которые вы там применили. Настройки там тоже бывают разные. Мы дальше, может быть, тоже о них поговорим. То есть тот метод, который мы использовали, он не требует больших вычислительных мощностей. Возможно, они потребуются в будущем. Это уже будет зависеть от задачи, которую мы будем решать. Сейчас задача достаточно, можно сказать, сложная, но достаточно и простая относительно. И тех ресурсов, которые есть, их достаточно. И достаточно вычислительных мощностей.

Александр Павлович Еремеев:

Примеры задач. Одно дело использовать для подготовки курса, другое дело для разработки какого-то продукта серьезного, промышленного или коммерческого.

Владимир Владимирович Чистяков:

Дальше мы это обсудим. Если вы не против, дальше уже будет с примерами, со всем этим. То есть мы дальше уже подробнее в это углубимся.

Роман Сергеевич Куликов:

Я чуть-чуть добавлю. Мы, естественно, заранее не знали оценок подробных. как по вычислительной мощности, но и тоже это очень условные получаются довольно метрики. Но сразу изначально мы ориентируемся решать задачу на той вычислительной мощности, которая в корпоративном формате доступна. Я думаю, что мы выйдем за пределы отдельного персонального компьютера, это будет какой-то небольшой там кластер корпоративного уровня. Вот в таком масштабе мы задачу ориентированно решать. И есть ощущение, что этого хватит. Не потребуется суперкомпьютер из рейтинга топ-100 компьютеров мира. Потребуется кластер корпоративного уровня, который сейчас в любой маломальской организации имеется, и он им по карману и так далее. Наверное, он должен быть адаптированным под эти задачи, то есть на видеоускорителях, то есть не на процессорах общего назначения.

Константин Александрович Петров:

Вот там четвертая снизу строчка «В зависимости от лимита контекста». И для RAG метода при больших документах часть данных теряется. А можно пояснить это вообще как? Каков механизм потери части данных?

Владимир Владимирович Чистяков:

Это зависит от количества размера входного окна. То есть если у вас слишком большие данные в RAG, то он просто их не сможет обработать и выдаст вам неправильное утверждение. В этом как бы и проблема.

Олег Вадимович Глухов:

Я объясню смысл. Вот если, допустим, мы берем размер контекста 500 токенов, то это примерно абзац, для понимания. Размеры одного запроса. Значит, у большинства моделей сейчас есть некое ограничение на размер вот этого входного контекста. Он примерно 128 тысяч токенов. Ну, то есть это, наверное, страниц 50 текста. А чем он ограничен? Вычислениями. То есть они специально так оптимизируют для того, чтобы, видимо, не сильно ждать долго ответ.

Роман Сергеевич Куликов:

Ты вручную можешь его увеличить?

Олег Вадимович Глухов:

Нет.

Степан:

У нас в RAG ограничение по размеру датасета или по размеру контекстного окна?

Олег Вадимович Глухов:

Контекстному окну. То есть, смотрите, допустим, мы даем запрос, проанализируем мне книгу какого-то тома «Война и мир». Соответственно, всю книжку мы просто не можем туда дополнительным контекстом дать, потому что RAG – это буквально дополнение твоего контекста информации. Все равно, что ты ей даешь какой-то справочник, который она подсмотрит в дополнение к твоему вопросу.

Если мы говорим про код, то для кода всегда эффективнее будет дообучение. Значит, RAG применяется только для какой-то чувствительной информации, где нельзя интерпретацию выводить. То есть буквально информация, которую надо передать. Как курс доллара или рубля.

Роман Сергеевич Куликов:

То есть нельзя ошибаться. Отлично. Я ответ на свой вопрос получил.

Владимир Владимирович Чистяков:

Здесь мы будем уже углубляться немного в наш метод дообучения модели. Начнем с того, что рассмотрим полносвязанную нейронную сеть. Каждый слой состоит из N нейронов, полностью соединенных с N нейронами следующего слоя. Грубо говоря, условно интерпретировано, так выглядит нейронная сеть, матрица, весов нейронной сети. И что мы делаем в этой сети? написали метод Lora, Qlora что этот метод из себя представляет то есть у нас есть вот эта матрица которая тоже вот дипсик допустим 1 3 миллиарда параметра мы их замораживаем и добавляем свои адаптеры. Мы создаем свою матрицу и добавляем к этому дипсик, то есть мы сам дипсик не меняем вообще никак не меняем мы просто добавляем свои обученные новые слои

Олег Вадимович Глухов:
Да, коллеги, если что-то непонятно — скажите, я постараюсь объяснить по-другому.

Роман Сергеевич Куликов:
То есть, новые слои добавляются?

Олег Вадимович Глухов:
Нет, здесь нет. Подходов несколько. Например, в YOLO действительно добавляются новые слои. А в нашем случае всё немного иначе. Представьте задачу оптимизации: у нас есть матрица весов, и мы с помощью градиентного спуска постоянно вносим небольшие корректировки — «добавочки».

Роман Сергеевич Куликов:
То есть нейроны остаются теми же?

Олег Вадимович Глухов:

Да. Можно сказать, что дельта-W — это как будто бы небольшая поправка, вычисленная на новой итерации. Задача метода — определить, чему равна эта добавка, и прибавить её к весам на выходном слое, который формирует финальный сигнал. Для этого используется аппроксимация, о которой рассказывал Владимир.

Владимир Владимирович Чистяков:

Да, можно представить это так: у вас есть завод, по трубам которого течёт обычная вода. А вы хотите получить сладкую — добавляете сироп, краситель, ароматизатор. То есть сама система (трубы) остаётся прежней, вы лишь добавляете компонент, изменяющий результат.

Роман Сергеевич Куликов:
То есть дельта-W состоит из двух матриц — B и A? Это как бы продолжение обучения, но уже на узкоспециализированном наборе данных?

Олег Вадимович Глухов:

Да, можно сказать, что градиентный спуск или сама задача оптимизации «вшита» в архитектуру. Формально — да, добавлен слой, но он работает параллельно. Представьте, у нас есть модель DeepSeek со 100 слоями — своего рода конвейер: вход, выход и промежуточные слои. Мы добавляем к ней параллельный «чёрный ящик», который вычисляет вот эту добавку к выходу модели.

Роман Сергеевич Куликов:
Ну, не совсем — это применяется не только к выходу, а вообще ко всем весам.

К выходному слою.

Олег Вадимович Глухов:

Да, то есть мы должны скорректировать. Но она на самом деле аппроксимирует всю эту нейросеть.

Роман Сергеевич Куликов:

Всю ее матрицу. Выходной слой не может всю матрицу аппроксимировать.

Там схема была. Ты говоришь 100 слоев. Да. Там миллиард параметров между всеми нейронами всех слоев. Какие-то связи есть. Они фиксируются в виде матрицы, но функции деактивации не менее их. Так? Да. Хорошо. Вот ты скачал дипсик. с этой матрицы и с этими функциями активации. Так? Так. Ты какие-то слои добавляешь или нет при дообучении?

Олег Вадимович Глухов:

В данном случае здесь используется программная настройка, которая это делает автоматически.

Роман Сергеевич Куликов:

Хорошо. Либо меняется, либо не меняется. Если не меняется, то матрица остается той размерности. Да. И если я не очень понял эти картинки, они не очень понятные. Откуда они взяты? Они выходят от Дельта В – это добавление к имеющейся матрице, просто новой матрице, то есть ее разметность та же самая, просто меняются значения в матрице.

Да, структура нейросети та же самая. Свои нейроны, их числа не меняются, меняются только весовые коэффициенты в матрице.

Олег Вадимович Глухов:

Ну, фактически, да.

Роман Сергеевич Куликов:

Да, структура нейросети та же самая. Свои нейроны, их число не меняется, меняются только весовые коэффициенты в матрице.

Олег Вадимович Глухов:
Ну, фактически, да.

Роман Сергеевич Куликов:
И тут, наверное, путает то, что… В процессе обучения структуру нейросети не меняют. В процессе обучения меняют весовые коэффициенты: дают проходить входным векторам на вход, смотрят, что на выходе возникает, сравнивают с тем, что желательно, формируя сигнал ошибки, используют его для обратного распространения или каким-то ещё способом корректируют веса в нейросети, не меняя число слоёв и число нейронов.

Олег Вадимович Глухов:
Да, но здесь ещё, смотрите, написано — «замороженная матрица». То есть мы будто бы берём некую заготовку и меняем её только на определённую дельту. Если просто вспомнить любую оптимизационную задачу, когда нам нужно оценить вектор параметров...

Владимир Владимирович Чистяков:
Да, формула — типа ΔW. Это сверху раскрыто

Александр Павлович Еремеев:

Можно еще маленький вопрос? Возможно, я не расслышал. Когда вы добавляете в замороженную матрицу дополнительную информацию, вообще может возникнуть такая проблема, что новая информация противоречит той, которая уже введена. Например, там введены, условно говоря, все птицы летают, а вы вводите пингвина, который не летает. Делается какая-то проверка или системой базовой, или... Потому что если там вошли противоречия, то система может получить какой угодно ответ. То есть и А, и не А. Вот эту задачу вы решаете или вы заведомо считаете, что здесь новая информация заменит старую. Но поскольку я вижу матрица заморожена, значит там ничего не заменится. Будет просто добавлена новая информация. А вот если она противоречит то, что уже есть в этой матрице. Что в этом случае делается? И вообще решается ли эта задача?

Олег Вадимович Глухов:

Проверка на противоречия сейчас не происходит. То есть мы такую задачу пока не ставили.

Александр Павлович Еремеев:

Тогда вы не гарантируете качество ответа. Система как будет решать? Она может взять одну, может взять другую. Или она заведомо берет то, что добавлено в случае противоречия.

Владимир Владимирович Чистяков:

Я даже вам так скажу, что модели, которые обучаются от GPT и так далее, они же обучаются с данных с интернета, там тоже очень много противоречий.

Александр Павлович Еремеев:

Нет, в GPT там немножко другое. Там есть, ну не знаю, это были предыдущие версии модели Wolfram, где специальные люди, администраторы высшего уровня, могут скорректировать информацию в GPT, заменить одну на другую. Обычный пользователь это не может. Но отсюда возникают и неверные ответы, и какая-то из причин, одна из галлюцинаций.

Павел Юрьевич Анучин:

Отвечая на вопрос, могу сказать, что здесь как бы особенность подхода именно в том, чтобы использовать оптимальные данные изначально. То есть замороженная матрица, она у нас будет, если нам важно работать с пингвинами и птицами, различать их корректно, то здесь именно часть подхода в том, что мы как тот разработчик, который будет такую модель дообучать в целях, опять же, птиц определять и так далее, мы будем использовать заведомо ту матрицу, которую предполагаем, что на это изначально и обучали. То есть она будет с птицами работать хорошо, то есть она не будет всеобъемлющим GPT или Дипсиком, но мы будем знать точно, что да, она небольшая, но ее под птиц обучали, и мы ее обучаем направленно.

Владимир Владимирович Чистяков:

Да, мы подходим к концу. На двадцатом слайде — как раз про это. Смотрите, почему вообще используют QLoRA. В чём смысл? Этот метод подходит для слабых вычислительных систем. Почему? Потому что он позволяет снизить нагрузку на железо — за счёт квантования, то есть уменьшения точности данных. Мы немного теряем в качестве, но при этом модель всё равно показывает вполне приличные результаты. Поэтому если вычислительные ресурсы ограничены, QLoRA — хороший вариант. На примере видно, что она просто квантует входные данные.

Олег Вадимович Глухов:

Да, верно. Она квантует именно матрицу весов — ту самую «замороженную» часть модели. Ведь даже несколько миллиардов параметров могут сильно нагружать процессор или видеокарту. При квантовании мы уменьшаем разрядность весов — например, вместо 16-битных значений (float16) используем 4-битные или 8-битные. Это и есть аппроксимация — компромисс между точностью и скоростью.

Шамиль Алиевич Оцоков:
А изначально какая разрядность используется в весах?

Олег Вадимович Глухов:
Обычно половинная точность — float16.

Владимир Владимирович Чистяков:

Да, может быть float32 или float16. А при квантовании мы уменьшаем до 4 бит, иногда до 8 — но мы решили сразу до 4, чтобы сильнее сократить объём. Поэтому модель в «сжатом виде» замораживается, а дальше уже происходит обучение поверх этой квантованной основы.

Есть компьютер, заходим на сайт, выбираем открытую модель, которая подходит под наши ресурсы. Каждый знает свои вычислительные возможности, поэтому подбирает сам. Потом просто скачиваем модель — там прямо приведён пример кода, который нужно вставить и запустить, чтобы загрузка пошла автоматически. Это базовый процесс.
Дальше создаётся датасет — и вот это уже гораздо более трудоёмкий этап. Подготовка данных занимает больше времени, чем само обучение. После этого пишем код, добавляем LoRA, настраиваем количество эпох, learning rate, шаг итераций — и запускаем обучение. В итоге получаем первые результаты.
Пока что мы не добавляли метрики вроде графиков потерь (loss) и других, но в следующей версии демо — 0.2 — они уже будут, чтобы можно было визуально отслеживать, как обучается модель.

Олег Вадимович Глухов:

Вот эти параметры, которые прописаны в конфигурации LoRA, как раз и задают ту самую дельту ΔW — изменения весов, которые мы должны рассчитать на нашем датасете и добавить к замороженной матрице.

Роман Сергеевич Куликов:
А вот тут выбираем нужную модель?

Владимир Владимирович Чистяков:
Да, потому что моделей много, в том числе специализированных. Есть и готовые датасеты — часть выкладывают компании, часть делают энтузиасты.

Олег Вадимович Глухов:

Да, на самом деле, по этой теме можно отдельный семинар провести — как вообще классифицируются модели на Hugging Face. Сейчас, последние полтора года, подход примерно одинаковый: любую современную модель сначала обучают в режиме обучения с учителем — ей подают огромный массив размеченных данных, и на этом формируется предобученная матрица.
После этого идёт этап дообучения с обратной связью — как раньше делали в GPT. Человек задаёт запрос, получает ответ и оценивает его: «палец вверх» или «палец вниз». Это и есть обучение с подкреплением, только с участием человека.
Иногда, вместо человека, эту обратную связь формируют другие модели — они анализируют ответы и решают, хороший он или нет.
На Hugging Face можно найти модели, которые как раз дообучались таким способом — большие системы вроде DeepSeek с сотнями миллиардов параметров оценивали результаты меньших моделей и «передавали» им свои корректировки.

Многие компании, кстати, публикуют свои модели, заявляя о высоких результатах — точности, полноте, низкой перплексии и прочих метриках. А потом выясняется, что они просто дообучили уже готовую большую модель, например тот же DeepSeek, и выдали за полностью новую разработку. Это вызывает споры — кто на самом деле автор и что именно сделано с нуля.

Владимир Владимирович Чистяков:
Я раньше вообще не понимал этот «птичий язык».

Роман Сергеевич Куликов:
Так и называется.

Владимир Владимирович Чистяков:
Ну то есть есть модели, которые, условно говоря, «размышляют», но сами ответы не выдают.

Роман Сергеевич Куликов:
Вот сайт — HuggingFace.com.

Павел Юрьевич Анучин:
Расскажите, кстати, об их идеологии.

Олег Вадимович Глухов:
Домен у них общий, открытый. Там можно выкладывать всё, что угодно. Это отличная площадка — любой желающий может публиковать и использовать модели.

Роман Сергеевич Куликов:
Коллеги, а сколько там вообще моделей? Порядка тысячи?

Олег Вадимович Глухов:
Нет, десятки тысяч.

Роман Сергеевич Куликов:
Десятки тысяч? И что, все они обучены с нуля?

Олег Вадимович Глухов:
Конечно нет. С нуля — только ограниченное количество, буквально несколько десятков. Это базовые крупные модели, а все остальные — их дообученные версии под конкретные задачи.

Александр Павлович Еремеев:
Да, был же тезис, что сделать новую модель с нуля — это дорого и долго.

Олег Вадимович Глухов:
Верно.

Роман Сергеевич Куликов:
А на HuggingFace выложены модели, которые откуда вообще взялись? Это fine-tuning тех же DeepSeek и других?

Олег Вадимович Глухов:
Да, часто именно так. Иногда это дообучение на других данных, иногда просто адаптация. У каждой модели на сайте есть страница — с описанием, кто автор, на чём обучалась, от какой модели произошла.

Роман Сергеевич Куликов:
А проверить это можно? Вот, допустим, я взял DeepSeek, немного его дообучил и заявил, что обучил свою модель «с нуля в Сибири».

Олег Вадимович Глухов:
Такое уже бывало. Были случаи, когда заявляли, что модель оригинальная, а потом оказывалось, что она просто обучена поверх более крупной. Это не преступление, но вводит в заблуждение.

Роман Сергеевич Куликов:
То есть большинство моделей на сайте — не с нуля?

Олег Вадимович Глухов:
Да, конечно. Это крайне редкое исключение. Ведь создать большую языковую модель с нуля — это миллионы долларов и огромные вычислительные ресурсы. Раньше такие проекты делали только крупные компании, и они не спешили делиться результатами.

Роман Сергеевич Куликов:
А DeepSeek стал первым, кто выложил такую модель в открытый доступ, верно?

Олег Вадимович Глухов:
Да, были и другие, но именно DeepSeek произвёл настоящий фурор.

Роман Сергеевич Куликов:
Хорошо. Значит, на *Hugging Face* в основном выложены так называемые «зафайнтюненные» модели — дообученные версии крупных открытых систем вроде DeepSeek, верно? А какие вообще есть открытые модели?

Владимир Владимирович Чистяков:
Из открытых — Qwen? *Mistral* — открытая. Ещё *LLaMA*.

Олег Вадимович Глухов:
Да, LLaMA.

Роман Сергеевич Куликов:
То есть открытых моделей — буквально несколько десятков?
И это действительно самобытные модели, не дообученные?

Владимир Владимирович Чистяков:
Да, примерно так.

Роман Сергеевич Куликов:
И при этом люди выкладывают их в открытый доступ, хотя потратили на разработку огромные деньги. Почему? Зачем?

Олег Вадимович Глухов:

Точных ответов нет, но можно предположить: компании не хотят выпадать из гонки open source. Они выкладывают не самые топовые версии своих моделей, чтобы оставаться в игре и привлекать внимание к себе.

Роман Сергеевич Куликов:
То есть, по сути, это обрезанные версии — «срезы» от полноценных моделей?

Олег Вадимович Глухов:
Скорее всего, да. Я бы, честно говоря, сам так и делал.

Роман Сергеевич Куликов:

Понятно. Эти урезанные модели потом берут другие разработчики, дообучают под конкретные задачи, и появляется масса вариаций.
Только вот возникает вопрос доверия — ведь непонятно, кто конкретно и как это делал. Я ведь могу скачать модель, дообучить её непонятно на чём и выложить. А кто-то другой потом будет использовать мои результаты.

Олег Вадимович Глухов:
Да, риск есть. Но в open-source сообществе работает условный «антивирус»: сотни людей тестируют модели, оставляют отзывы и находят проблемы.

Роман Сергеевич Куликов:

Ну, формально да. Но по факту никто нормально обратную связь не даёт — большинство отзывов формальные или управляемые. Нормальные люди не пишут отзывы ни о ресторанах, ни об open-source проектах.

Павел Юрьевич Анучин:

Да, и тут есть интересный момент. Когда крупные компании выкладывают модели, это не только ради open-source, но и способ спровоцировать других игроков. Например, если ты — DeepSeek, тебе интересно, что под капотом у GPT. Они долго ничего не публиковали. Тогда ты выкладываешь свою модель, показываешь результат — и тем самым как бы «заставляешь» их ответить, тоже что-то выложить. Это такая мягкая провокация.

Олег Вадимович Глухов:
Да, согласен. Обычно выкладывают не самую большую модель, а что-то вроде облегчённой версии — на десятки миллиардов параметров. Но и этого достаточно, чтобы вызвать интерес. А держать в открытом доступе модель на триллионы параметров смысла нет.

Роман Сергеевич Куликов:

В целом понятно — самые передовые технологии никто не выкладывает, но даже то, что есть, уже огромный шаг.

Олег Вадимович Глухов:
Верно. Главное, что теперь появилась высокая доступность. Даже индивидуальные разработчики, студенты и школьники могут экспериментировать с языковыми моделями.

Роман Сергеевич Куликов:

Да, ещё пару лет назад это было невозможно.

Олег Вадимович Глухов:
Это как библиотека OpenCV — готовые функции, бери и используй.

Роман Сергеевич Куликов:
Да, всё верно. Ну что ж, дальше — про датасеты. Здесь есть один пример.

Владимир Владимирович Чистяков:

Да. Сначала немного о технических деталях. Наш первый датасет, если честно, не идеален — мы просто хотели быстро проверить, получится ли вообще что-то осмысленное.
Сделали примерно тысячу примеров, чтобы протестировать сам процесс. Цель была — обучить модель писать на *SystemVerilog* в определённом стиле кода.
Но, как оказалось, модель *SystemVerilog* не знала — только обычный *Verilog* и *VHDL*. Они похожи, но отличаются по синтаксису.

Олег Вадимович Глухов:
Можно обратить внимание, как называлась модель — *DeepSeek Coder 1.3B Instruct*.
То есть это модель для генерации кода, обученная в режиме инструкций — «кодер» и «инструктор».

Владимир Владимирович Чистяков:

Да. «Coder» — значит, что она заточена под написание кода, а «Instruct» — что она воспринимает инструкции в явном виде, через промпт.
Есть, например, модели Think — они просто рассуждают и не дают прямого ответа. А Instruct-модель получает инструкцию и выдаёт конкретный результат.
Вот я для теста попросил её: «Напиши мультиплексор с параметром ширины данных». Мы специально не указали ширину — хотелось посмотреть, что она выберет сама. И модель выдала код на Python. Не совсем то, конечно. Поэтому мы уточнили задачу: «Напиши мультиплексор на SystemVerilog».

Владимир Владимирович Чистяков:

Как видно, модель не совсем справилась с задачей. Во-первых, в коде нет управляющих сигналов — то есть вход не используется как управляющий. Во-вторых, она написала код на Verilog, а не на SystemVerilog. В общем, не совсем корректно.

Олег Вадимович Глухов:

Понял. Надо смотреть последовательность — от левого края, да?

Владимир Владимирович Чистяков:

Да, именно так. Я просто поставил многоточия, потому что код получился слишком длинным и не помещался на слайд. Не хотел уменьшать шрифт, поэтому сделал вырез. Наверное, лучше было оформить это таблицей.

Роман Сергеевич Куликов:

Ну ладно.

Владимир Владимирович Чистяков:

Дальше уже пример с нашей моделью — той, которую мы обучили на собственных промптах. Это уже обученная версия. Она переняла стиль написания — camelCase, о котором я говорил. Да, параметр ширины данных она выбрала немного завышенный, но в целом результат уже близок к ожидаемому.

Интересно, что теперь модель чаще всего выдает код именно на Verilog, VHDL или SystemVerilog, если явно указать язык. Раньше она иногда писала на Python, но после обучения — больше нет.

И ещё — она стабильно использует camelCase даже на других языках, что говорит о закреплении стиля.

Роман Сергеевич Куликов:

А теперь последний слайд — плюсы и минусы.

Владимир Владимирович Чистяков:

Да. В целом модель удалось обучить. Мы увеличили датасет с тысячи до двенадцати тысяч примеров.

CamelCase — это, если коротко, когда первое слово пишется с маленькой буквы, а каждое следующее — с большой, например dataADC. Мы используем этот стиль, потому что он делает код более читаемым, особенно когда проект большой — по сто-двести строк. Сразу видно, где вход, где выход.

Да, модель тоже переняла этот подход. Обучение заняло около сорока минут — не потому что процесс сложный, а из-за небольшого объема данных и упрощений. Сейчас мы планируем дообучить модель на более качественном датасете, без таких упрощений. Это будет вторая версия, более мощная и продуманная.

Олег Вадимович Глухов:

Да, и главное, что всё это — не какая-то магия. Обучение можно провести на обычном компьютере, без огромных вычислительных мощностей. Всё упирается в качество датасета.

Павел Юрьевич Анучин:

А можно уточнить: если в промпте не указывать, что нужно писать именно на SystemVerilog, модель может сгенерировать код на другом языке, верно? А как добиться того, чтобы она всегда писала именно на SystemVerilog?

Владимир Владимирович Чистяков:

Теоретически, можно полностью переобучить модель — это был бы самый надёжный способ. Но для разработчиков это вряд ли имеет смысл. У всех свой стиль: кто-то пишет на VHDL, кто-то на Verilog.

А вот если использовать модель в образовательных целях — для студентов, тогда стоит добиваться единого стиля, чтобы все писали одинаково. Как именно этого достичь, я пока не могу сказать точно — не пробовал. После тестов смогу ответить увереннее.

Павел Юрьевич Анучин:

Я видел инструмент, который называется system prompt — системный промпт. Он работает как фильтр: добавляется автоматически перед каждым запросом и влияет на ответ.

Олег Вадимович Глухов:

Да, всё верно. Но как раз из-за того, что системный промпт не всегда помогает, и прибегают к дообучению. Он просто объединяется с твоим текстом запроса под капотом. Иногда работает, но часто — нет, особенно при генерации кода. Поэтому обучение на специализированных данных надёжнее.

Владимир Владимирович Чистяков:

Вопросы есть?

Роман Сергеевич Куликов:

А какая базовая модель вообще использовалась?

Олег Вадимович Глухов:

Мы взяли DeepSeek.

Роман Сергеевич Куликов:

То есть, если я правильно понимаю, ты взял базовую модель и дообучил её под конкретные задачи?

Владимир Владимирович Чистяков:

Да, именно так. Это был исходный DeepSeek.

Олег Вадимович Глухов:

Верно — голый DeepSeek, просто в уменьшенной версии для локального использования.

Роман Сергеевич Куликов:

И он уже умел писать код на Verilog?

Владимир Владимирович Чистяков:

Да, но не идеально. Насколько я помню, это была модель DeepSeek Coder 1.3B, специально заточенная под программирование.

Роман Сергеевич Куликов:
То есть получается, какой-то индус обучил свой же DeepSeek и выложил его.

Владимир Владимирович Чистяков:
Ну, примерно так. Модель изначально обучалась писать код, возможно — с нуля.

Роман Сергеевич Куликов:
Вы уверены, что с нуля?

Владимир Владимирович Чистяков:
Не факт. Может, и нет. Никто точно не знает.

Роман Сергеевич Куликов:
То есть где-то в недрах DeepSeek создали эту новую версию DeepSeek.

Владимир Владимирович Чистяков:
Да.

Роман Сергеевич Куликов:
Хорошо. А ты, значит, дообучал именно её?

Владимир Владимирович Чистяков:
Да, всё верно.

Роман Сергеевич Куликов:
То есть она могла быть создана с нуля, а могла быть просто дообучена. Вероятнее, второе — учитывая, какие ресурсы нужны для полного обучения.

Олег Вадимович Глухов:
Да, согласен. И здесь стоит упомянуть про метод LoRA или QLoRA — он сейчас самый популярный. Его как раз используют для дообучения без больших вычислительных затрат.
Если бы DeepSeek Coder дообучался этим методом, мы бы это увидели в структуре модели: там появляются дополнительные прослойки — адаптеры.

Роман Сергеевич Куликов:
То есть это такой внешний «надстройка», которую при желании можно отсоединить?

Олег Вадимович Глухов:
Да. Если её убрать, останется «замороженная» базовая модель. Разве что кто-то специально зашьёт изменения прямо в бинарный код, чтобы их скрыть. Но обычно всё видно.

Роман Сергеевич Куликов:
А после дообучения получается ведь та же матрица весов?

Олег Вадимович Глухов:
Фактически две. Есть исходная матрица и добавочная. Они хранятся как отдельные файлы.

Роман Сергеевич Куликов:
Понял. Посмотрю подробнее.

Олег Вадимович Глухов:
Да, можно. Главное, что дообучение — это по сути просто математическая операция. Весовую матрицу скачиваешь как обычный файл.

Владимир Владимирович Чистяков:
Ну, у нас основная цель работы была в другом — проверить, можем ли мы вообще на обычном компьютере обучить модель хоть чему-то. И, в принципе, смогли.

Роман Сергеевич Куликов:
Да, понятно. Мы ведь не корпорация, а университет, здесь подход исследовательский.
Согласен.

Владимир Владимирович Чистяков:

Что касается других моделей — пробовали и более крупные версии, вроде 10B или 11B. Они код пишут, но не идеально. Маленькие примеры — нормально, большие уже нет.

Олег Вадимович Глухов:
А это специализированные модели?

Владимир Владимирович Чистяков:
Нет, просто проверял из интереса — запускал локально, смотрел, как работает.

Роман Сергеевич Куликов:
А как оцениваете качество работы модели? Качество кода, который она генерирует?

Владимир Владимирович Чистяков:
Пока я просто проверяю вручную — задаю задачу и смотрю, правильно ли написано. Автоматических метрик пока нет.

Роман Сергеевич Куликов:
А данные для обучения откуда брались?

Владимир Владимирович Чистяков:

Часть написал сам, часть взял из открытых материалов. В итоге получилось около тысячи примеров.
Например, я давал промпт вроде: «Напиши мультиплексор 4 к 1 с параметрами» — и использовал эталонные решения из методичек.
Брал не всё подряд, а те, где примеры корректные, понятные, с пояснениями.

Роман Сергеевич Куликов:
То есть использовалась наша учебная методичка?

Владимир Владимирович Чистяков:
Да, частично. Что-то из неё, что-то из курсов и открытых датасетов, где было нужно — переделывал под свой стиль.
Датасетов именно по SystemVerilog почти нет, в основном — по Verilog и VHDL. Есть наборы, где обучают переводу между ними, но не по самому SystemVerilog.

Роман Сергеевич Куликов:
Да, то, что нашли, то и использовали.
Но вообще интересно — если взять десяток хороших учебников по программированию ПЛИС, там ведь обычно всё выстроено дидактически — от простого к сложному, с примерами.

А насколько сейчас вообще можно выстроить процесс обучения модели по этому принципу?
Не просто «ткнули пальцем и посмотрели, что выйдет», а именно пошагово — от простого к сложному, как учат студентов.

А если подходить системно — как к обучению человека?
То есть от самых азов: берём первую главу учебника, закрепляем материал, проводим зачёт, экзамен. Пока не освоено — не двигаемся дальше. Такой подход оправдан? Или всё же лучше просто оцифровать весь учебный материал и обучать модель сразу на полном объёме?

Олег Вадимович Глухов:

Да, я бы сказал, второй вариант выглядит предпочтительнее. Нейросеть ведь не человек — ей не нужно проходить материал последовательно. Ей, наоборот, полезно видеть разнообразные данные, разного уровня сложности. Это повышает способность к обобщению. Поэтому чем больше разнородных данных она увидит на старте, тем лучше.
А если есть электронные версии — и вовсе идеально, даже оцифровывать ничего не придётся.

Владимир Владимирович Чистяков:
Не важно, в каком виде, главное — чтобы данные можно было подать.

Роман Сергеевич Куликов:
Да, тогда алгоритм такой: собрать всё, что есть по теме, — купить, скачать, объединить в один большой архив, запустить обучение и на время уйти на рыбалку. Пусть модель сама «переваривает». А потом вернуться и проверить результат.

Олег Вадимович Глухов:
Да, только обязательно нужна методика проверки. Не просто примеры «в лоб», а полноценная система оценки качества на большой выборке.

Владимир Владимирович Чистяков:
Вот, и тогда логика следующая: собрали весь материал, провели цикл обучения, получили результат — и передали экспертам. Они проверяют, где хорошо справилась, где нет. Потом вносятся корректировки, и запускается следующий цикл.

Олег Вадимович Глухов:
Согласен. Тем более сейчас активно применяются методы «заморозки весов» на разных этапах — после первой, второй, третьей эпохи можно остановить обучение, посмотреть качество, подкорректировать и продолжить. Это экономит кучу времени — не нужно ждать, пока закончится сотня эпох.

Роман Сергеевич Куликов:
Да, то есть можно контролировать процесс обучения в реальном времени. Но тут возникает вопрос: стоит ли обучать нейросеть сразу на всём массиве знаний или всё-таки пошагово, как человека — от темы к теме?

Александр Павлович Еремеев:
Я считаю, что постепенно. Пошагово и с промежуточной проверкой результатов.
Если учить всё сразу, возникает риск переобучения — модель запоминает материал, но теряет способность к обобщению. Поэтому важно контролировать каждую эпоху и отслеживать, как она усваивает материал.

Роман Сергеевич Куликов:
Вот, то есть остаётся два подхода: как человека — от простого к сложному, или как нейросеть — сразу на всём объёме данных.
Олег, ты сторонник второго варианта — собрать сотню книг и обучать модель сразу на полном наборе знаний. Но все ли с этим согласны? Или кто-то считает, что надо идти постепенно, глава за главой?

Александр Павлович Еремеев:

Разрешите добавить.
Когда мы говорим о рассуждающих системах и логическом выводе, всё начинается с аксиоматики — набора базовых истин, на которых строятся дальнейшие рассуждения.
Дальше система выводит новое знание из старого, опираясь на правила логического вывода — чтобы из корректного получалось корректное.
Причём это могут быть вероятностные модели, где вывод не гарантирован, но правдоподобен.
Простой пример — аксиоматика веры: если человек верит в Бога, он выстраивает одни рассуждения; если не верит — другие.

Так и здесь: важно сначала задать правильную основу — ту аксиоматику, которую мы считаем достоверной. Тогда поверх неё можно строить модели рассуждения, включая современные имитаторы человеческой логики.

Я вспоминаю конференцию по искусственному интеллекту, где гуманитарии из Плехановки приводили пример: студенты, скачивающие рефераты по новейшей истории, получают искажения — потому что GPT обучен на других базах данных, где исторический контекст иной.
Поэтому, когда мы говорим о машинном обучении, надо поэтапно проверять, чтобы новые данные не противоречили базовому контексту системы.

Олег Вадимович Глухов:

Согласен. Но уточню: вопрос не в процессе обучения, а в формировании датасета. Стоит ли сразу включать в него весь уровень знаний — от базового до продвинутого, или сначала обучать на азах, проверить качество, а потом добавлять материал посложнее?

Александр Павлович Еремеев:

Да, хороший вопрос.
Есть известная аксиома: чем универсальнее система, тем менее эффективна в конкретной задаче.
Самый универсальный метод — полный перебор, но он практически нереализуем.
Американцы пошли по пути GPT — гигантские матрицы, терабайты данных, все области знаний.
Китайцы (на примере DeepSeek) сделали иначе — построили иерархическую систему, где верхний уровень классифицирует задачу и направляет её на нужный «модуль» или агента, специализирующегося на своём типе задач.
Это фактически мультиэкспертная архитектура: сверху классификатор, ниже специализированные эксперты.
Такой подход ближе к системному анализу и иерархии Саати — более рациональный и масштабируемый.

Роман Сергеевич Куликов:
То есть вопрос — стоит ли такую иерархию применять и в нашем случае?
У нас ведь уже есть большая языковая модель, и мы хотим научить её специфике ПЛИС-разработки.
Так вот, обучать ли её всему сразу — всем знаниям о ПЛИС, или тоже дробить по уровням сложности?

Александр Павлович Еремеев:

Если загнать всё в одну систему, появится неопределённость.
Лучше, чтобы модель умела различать тип задачи — например, естественно-языковую, математическую, инженерную.
Врач общей практики ставит диагноз и направляет к нужному специалисту — вот аналог.
Так же и здесь: универсальный «мозг-демиург» звучит красиво, но неэффективен. Лучше иметь иерархию — базовую модель и специализированные надстройки. Это проще, надёжнее и меньше противоречий.

Роман Сергеевич Куликов:

Не соглашусь. В нашем случае, программирование ПЛИС — область строго формализованная, без противоречий.
Там всё однозначно определено языком и архитектурой.

Александр Павлович Еремеев:

Да, конечно. Но даже в этом случае вы не будете включать туда знания из медицины или социологии.
Поэтому подход DeepSeek — логичен: есть общий уровень, а внутри — специализированные агенты для конкретных задач.

Олег Вадимович Глухов:
А если говорить в рамках одного агента — разработчика ПЛИС, то как его обучать?

Роман Сергеевич Куликов:
Можно по-разному. Например, как человека — от простого к сложному: сначала сделать счётчик, потом умножитель, потом ячейку памяти. И к следующему шагу переходить только после успешного освоения предыдущего.
А можно загрузить всё сразу — весь объём знаний по программированию ПЛИС, со всеми учебниками и примерами, и дать модели возможность усвоить всё за одну эпоху.

Александр Павлович Еремеев:

Здесь всё зависит от предметной области.
Если задачи однотипные и чётко структурированные — можно загружать всё сразу.
Если же есть значительные различия по классам задач — лучше делить обучение по уровням и специализациям.
Но в общем случае — чем уже сфера и конкретнее задачи, тем эффективнее работает специализированная модель.

Роман Сергеевич Куликов:
Ладно, вопрос снимаю, в целом понятно. Павел, кажется, конкретных ответов мы пока не получили.

Павел Юрьевич Анучин:

Да, я хотел уточнить. Получается, что конкретных плюсов и минусов у двух подходов — пошагового и полного обучения — мы пока явно не выделили, верно?
Может, кто-то может предположить, исходя из опыта или конкретных кейсов, где какой метод лучше себя показывает?

Константин Александрович Петров:

Я попробую ответить. Всё зависит от цели.
Если задача узкоспециализированная — например, мы обучаем модель под конкретную предметную область — то логично просто взять весь датасет и загрузить его целиком.
А если мы говорим о большой универсальной системе, которая должна работать с разными контекстами, языками, культурами — как, например, гуманитарные модели, — тогда без поэтапного обучения и постоянной верификации не обойтись.

Павел Юрьевич Анучин:

Понял. А если вернуться к нашей задаче — разработчик ПЛИС, узкая специализация. Есть ли смысл идти от простого к сложному, или лучше сразу загрузить весь датасет целиком?

Константин Александрович Петров:
Мне кажется, в этом случае нет большой необходимости дробить материал. Объём данных сравнительно небольшой, поэтому можно загрузить всё сразу и посмотреть результат.

Павел Юрьевич Анучин:
А с точки зрения методологии — даже если данных немного — даёт ли пошаговое обучение хоть какое-то преимущество в качестве?

Роман Сергеевич Куликов:

У поэтапного подхода есть плюс — можно контролировать процесс.
Ты видишь, как модель осваивает каждый уровень, где у неё ошибки, и можешь корректировать.
А если заливаешь весь датасет целиком, результат может оказаться случайным: непонятно, на каком этапе произошла ошибка, и где именно модель «поплыла».
С другой стороны, целостное обучение быстрее и требует меньше усилий.

Олег Вадимович Глухов:

Да, но у поэтапного подхода есть и обратная сторона.
Если мы будем учить нейросеть последовательно — сначала на «А», потом на «Б», потом на «В» — то, когда она перейдёт к новой информации, начнёт забывать старую.
Это типичная проблема — catastrophic forgetting.
Поэтому современные системы искусственного интеллекта обучаются иначе: весь датасет собирается сразу, и примеры подаются модели в случайном порядке.

Так сохраняется баланс между старым и новым опытом.
Если дробить обучение слишком мелко, то к моменту, когда дойдёшь до последнего блока знаний, от первых не останется следа.

Зурова Мария Александровна:

Позвольте короткое рассуждение.
Если представить стандартную, пусть даже сложную языковую модель, можно рассмотреть две задачи — простую и более сложную, являющуюся её естественным расширением.
Для человека очевидно, что вторая — это развитие первой. А для модели это могут быть вообще разные алгоритмы.
Если мы обучаем на большом датасете сразу, то она корректно решит обе задачи просто потому, что видела весь спектр примеров.
А если обучаем поэтапно, начиная с простых случаев, нужно заранее заложить систему знаний и рассуждений.
То есть это уже не просто языковая модель, а скорее система поддержки принятия решений — с явной аксиоматикой и базой правил.
Но нейросети ведь не хранят знания в явном виде.
Поэтому поэтапное обучение, аналогичное человеческому, требует хранить знания и наращивать их.
А здесь это фактически невозможно — и тогда сам подход теряет смысл.

Олег Вадимович Глухов:

Да, Мария, абсолютно верно.
Проблема в том, что матрица весов при обучении постоянно корректируется, и старые знания теряются.
А логические рассуждения, заданные явно, как раз позволяют «высечь в камне» те базовые принципы, которые не должны изменяться.
Это уже не чистая LLM, а гибрид — экспертная система плюс языковая модель.

Зурова Мария Александровна:

Да, именно. Я, в целом, сторонник таких гибридных систем.
Но если говорить о чистых нейросетях — им эффективнее обучаться сразу на всём объёме данных, без разбивки по шагам.

Олег Вадимович Глухов:
А как, на ваш взгляд, можно реализовать подобную гибридную архитектуру? Как заложить эти логические компоненты, чтобы они реально работали?

Зурова Мария Александровна:

Возможно, через систему агентов.
Один агент хранит базовые правила — аксиоматику, которую нельзя нарушать, например правила самого языка.
Второй — отвечает за элементарные логические рассуждения и процессы, вроде выбора оптимальной структуры цикла.
И уже поверх них можно обучать большую модель на полном датасете.
Перед выдачей ответа система будет «пропускать» результат через этих агентов-проверяющих — через агента языка и агента логических правил.

Олег Вадимович Глухов:
То есть такие агенты, по сути, позволят избавиться от галлюцинаций модели?

Зурова Мария Александровна:

Да, именно так. Это проверка на логическую согласованность вывода.
Как и в примере с кодом — модель может написать цикл, но его корректность нужно проверять.

Олег Вадимович Глухов:

Понятно. Тогда логично начать с большого датасета, собрать максимум учебных материалов и посмотреть, справится ли модель.
Если обнаружатся галлюцинации — тогда уже добавлять прослойки экспертных агентов.

Александр Павлович Еремеев:

Позвольте ремарку.
Игорь Ашманов когда-то справедливо заметил: бум нейросетей не всегда оправдан.
Да, универсальные системы, способные ответить на всё, привлекательны.
Но если речь идёт, например, о системах поддержки оператора атомного энергоблока — там нужна гарантия корректности.
И здесь уместнее строить экспертные системы на базе продукционных правил, которые интерпретируются в дерево решений.
Такое дерево можно проверить на полноту, непротиворечивость, предсказать время отклика.
Нейронная сеть такого не даст — она выдаёт вероятностный ответ, без гарантии достоверности.
Если задача критическая — лучше экспертная система.
Если же задача сложная, плохо формализуемая — тогда да, нейросеть, но с пониманием ограничений её правдоподобия.

Олег Вадимович Глухов:
Это понятно. А есть ли примеры, где экспертную систему реально научили писать код?

Александр Павлович Еремеев:

Попытки были. Некоторые системы создавались с применением элементов ИИ, например для генерации музыкальных произведений или автоматизации программирования.
Но говорить о серьёзных промышленных решениях пока рано.
В простых задачах — да, работает. А в критически важных областях вопрос достоверности пока не решён.

Олег Вадимович Глухов:
Хорошо, понятно. Коллеги, есть мнения?

Челышев Эдуард Артурович:

Я добавлю.
Мы сейчас говорим про ассистента, который пишет код на Verilog.
Но стоит задаться вопросом — это конечная цель?
Или цель шире — создать универсального инженерного ассистента, который знает весь курс радиотехники, включая электротехнику, теорию передачи информации и т.д.?

Роман Сергеевич Куликов:
Да, именно, чтобы можно было охватить все учебные планы и направления.

Челышев Эдуард Артурович:

Вот именно. Ведь LLM, когда мы спрашиваем её «2×2», не считает, а просто выдает вероятностный ответ, который не обязательно равен 4.
Иногда лучше, чтобы модель просто вызвала калькулятор — выполнила вычисление детерминированно.
Так и здесь: LLM может быть «оркестратором» — управлять простыми модулями вроде калькуляторов, симуляторов и специализированных моделей.

Роман Сергеевич Куликов:
Да, нам и нужно, чтобы она могла пользоваться саппортом для моделирования СВЧ-эффектов, как инструментом.

Челышев Эдуард Артурович:
Вот именно — сначала калькулятор, потом саппорт, а LLM пусть управляет.

Роман Сергеевич Куликов:

Да, но саппорт не знает, какой формы сделать антенну. Он только считает заданную форму.
Поэтому нужен компонент, который будет решать творческую часть — выбирать форму элементов и конфигурацию. Сейчас это делает человек.

Олег Вадимович Глухов:

Понятно. Кажется, у всех выстраивается единое понимание.
Если говорить о большом инженерном ассистенте — гибридный подход действительно оптимален.
Но если задача уже узкая — например, просто код на системном Verilog — тогда вопрос: стоит ли подавать все учебники в датасете сразу, в случайном порядке, или идти от простого к сложному?

Роман Сергеевич Куликов:
Если честно, однозначно сказать сложно, но, скорее всего, первый вариант — полный датасет сразу — выглядит практичнее.

Олег Вадимович Глухов:

Да, аналогично тому, как учат распознавать изображения: не по одной букве, а сразу на всём алфавите.
Задача локальная, чётко определённая — так что случайная подача примеров по всей совокупности знаний выглядит разумной.

Роман Сергеевич Куликов:
Согласен. Это, пожалуй, наиболее реалистичный первый шаг.

Александр Павлович Еремеев:

Только не забывайте: нейросети не умеют объяснять, как они пришли к ответу.
А в обучении студентов важно понимать ход рассуждений.
Поэтому сейчас активно исследуют объяснимый ИИ — добавление объяснительных компонентов.

Олег Вадимович Глухов:
Да, это уже другая задача — педагогическая.
Мы сейчас всё же говорим о машинном обучении для инженерных задач.

Александр Павлович Еремеев:

Понимаю. Просто замечу, что при разработке важно избегать постоянного переобучения.
Когда система получает новые данные и сразу перестраивает веса, она начинает «забывать» старое и теряет устойчивость.
Поэтому нужно чётко понимать цель: это поисковое проектирование, цифровой двойник, анализ множества вариантов?
Под каждую цель — своя архитектура и метод обучения.
Отсюда и вывод: современные интеллектуальные системы должны быть интегрированными — сочетать разные подходы и адаптироваться под задачу.

Олег Вадимович Глухов:
Хорошо, спасибо. Коллеги, кто-то ещё хочет высказаться?

Роман Сергеевич Куликов:
Если нет, давайте кратко перейдём ко второму пункту.
Есть ли предложения по тематике следующего семинара?

Олег Вадимович Глухов:

Да, есть предложение: 3 числа не проводить, а перенести встречу на 17-е, то есть через месяц.
У многих в начале сентября совпадает с началом учебного года, поэтому удобнее чуть позже.
Если пока нет идей по теме — обсудим их в рабочем порядке.

Александр Павлович Еремеев:

Коллеги, было предложение заслушать Шамиля Алиевича — по его работе с DeepSeek.
Может быть, посвятим следующий семинар именно этому?

Олег Вадимович Глухов:
Шамиль Алиевич, сможете подготовить доклад к тому времени?

Шамиль Алиевич Оцоков:

Да, конечно. Я ещё попрошу одного студента, который занимался похожей темой, выступить вместе со мной.

Олег Вадимович Глухов:

Отлично. Тогда готовьтесь, я с вами свяжусь и уточню, будет ли это ближайший семинар или следующий.

Кроме того, коллеги из МИРЭА тоже хотели рассказать о своей платформе — они переносили доклад на сентябрь.
Есть ещё несколько идей, обсудим дополнительно.

Шамиль Алиевич Оцоков:
Да, через месяц удобнее — 3 сентября уже начинается учебный процесс.

Олег Вадимович Глухов:

Согласен. На 3 число не ориентируемся. Хорошо, коллеги, всем большое спасибо за участие. Всех ждём на следующем семинаре. Всего доброго!

Александр Павлович Еремеев:
Можно будет прислать сообщение, когда выложат доклады, чтобы мы могли их посмотреть?

Олег Вадимович Глухов:
Да, конечно. Уже на этой неделе всё появится на портале МИИ — там будут все презентации, включая материалы предыдущих семинаров. Их можно будет скачать и просмотреть.

Роман Сергеевич Куликов:
Спасибо. Всем до свидания.

Олег Вадимович Глухов:
До свидания, коллеги. Спасибо всем.