**Критерии оценки**

1. Метрики качества и производительности решения

Для оценки качества работы алгоритма будет использована метрика Mq :

,

где ,   
S — количество замененных слов,

D — количество пропущенных слов,

I — количество добавленных слов,

N — количество слов в исходном тексте.

F1\_weighted (F1-мера) — среднее гармоническое точности и полноты классификации, взвешенное по классам.

Для оценки производительности алгоритма будут использованы Latency\_score, задержка во время выполнения операции распознавания команды за секунду реального времени, и Memory\_score, пиковое потребление памяти (Мб) во время инференса алгоритма распознавания.

*У участников хакатона* ***нет необходимости*** *рассчитывать метрику для работы алгоритма самостоятельно. Необходимо выполнить расчет частных метрик качества распознавания (WER, F1-score) для тренировочного (создается вами из train части) и валидационного (создается вами из train части) наборов данных, производительности (Latency, Memory) алгоритма и отразить эту информацию на слайде. После стоп кода решения будут запускаться для отработки на тестовом датасете (создается постановщиком из данных, которых нет в train)*

2. Обоснованность выбранного метода.

*Участникам хакатона необходимо обосновать выбор методов распознавания. Во время защиты участникам требуется уделить бóльшее внимание описанию этапов работы алгоритма и архитектурным особенностям, преимуществам и ограничениям предложенного подхода.*

3. Масштабируемость решения

*Состав голосовых сообщений способен изменяться в будущем (уменьшиться или увеличиться). Участникам хакатона предлагается оценить потенциал и трудоемкость адаптации алгоритма под новые команды.*

4. Опыт взаимодействия пользователя с предложенным решением.

*Внешний шум и помехи приводят к потере качества входных данных, случайное нажатие может привести к неумышленной записи голосовой команды. Участникам хакатона предлагается проявить фантазию и предложить пути решения данных проблем.*

5. Адаптируемость и/или совместимость предложенного решения с языками программирования C, C++.

*Участники хакатона разрабатывают Python-прототип алгоритма распознавания голосовых команд. Алгоритм и/или модель должны иметь потенциал портирования на языки C, C++. Требуется обозначить степень сложности портирования алгоритма, самостоятельно портировать решение не требуется.*

**Выходные данные**

Программный код

*Участники хакатона оформляют свое решение в виде открытого GitHub-репозитория. Представителями компании-кейсодержателя будет производиться запуск решений на отложенной выборке данных. Для успешного запуска решения участникам хакатона рекомендуется:  
1) Самостоятельно оценить запускаемость решения.  
2) Подготовить описание структуры и работы решения, достаточного для восприятия и запуска подхода, в виде README.md файла.*

*3) Проверить актуальность использованных ссылок и доступность весов модели для запуска алгоритма.*

*4) Подготовить описание окружения развертывания решения. Допускается использование PDM, Poetry, conda, Docker окружений. Наличие файла requirements.txt с актуальными зависимостями обязательно.*

*5) Использовать скрипт get\_submission.py для чтения входных данных и приведения выхода алгоритма к требуемой структуре. Участники вправе добавлять собственные функции, классы и методы в предложенный скрипт, однако формат выходных данных должен соответствовать требуемой структуре.*

Пример выходных данных

Файл JSON, имеющий следующую структуру:

